FEDERAL STUDENT AID

We Help Put America Through School

FSA Modernization Partner

NSLDS II Reengineering Technical Architecture Plan

Version 1.0

July 19, 2002

Table of Contents

Document Control

Version Number	Description	Release Date	Author
1.0	Initial Issue	July 19, 2002	Justin Miller

1 Introduction

The National Student Loan Data System (NSLDS) was established as part of the Higher Education Act of 1965, as amended, to provide a comprehensive repository of information about Title IV recipients and their loans, grants, lenders, guaranty agencies, servicers and schools. Currently, NSLDS is hampered by a number of challenges related to discrepancies between the quality and timeliness of data feeds and the system of record, and its operating costs.

Given these challenges, the NSLDS II Reengineering project has been undertaken to improve financial and data integrity, reduce operational costs and improve customer satisfaction. NSLDS II will focus on the Data Warehousing and Internal FSA Direct Access opportunities as well as assessing ways to support existing requirements through NSLDS II or other modernized systems.

1.1 Purpose

The NSLDS II Technical Architecture Approach starts by building the context of the existing legacy NSLDS and then contrasting it with standard data warehouse architecture. Next, there is a brief overview of the existing tools and architectures present in the current FSA environments. This is followed by the planned new architecture for the NSLDS II starting with a review of the database vendor selection process. The new architecture is then outlined in terms of the data acquisition, storage, access and presentation technical architecture for the reengineered NSLDS II. Lastly, the development and execution architectures for the NSLDS II environment are detailed.

The initial phase of the NSLDS II reengineering effort involves replatforming the legacy NSLDS mainframe system onto a mid-range platform with the goal of achieving operational cost savings and improved access to data. The technical architecture has been planned with the following goals in mind:

- Maximize reliability
- Maximize efficiency
- Minimize change for existing interface partners

As a comprehensive data store for federal student lending data, NSLDS II will need to interface with a number of external systems to maintain current data. To meet the large data storage needs, a robust database solution is required to house the data that NSLDS II will receive, both as a part of the data conversion effort and as a part of ongoing operations. Lastly, NSLDS II will require a new data access component to retrieve data and present it to the end user.

1.2 Approach

The following approach was used to develop the NSLDS II Reengineering Technical Architecture deliverable:

- Review the data and documentation regarding legacy NSLDS
- Review and modify existing FSA best practices to meet the NSLDS II interface requirements
- Document the process required for applications to integrate and utilize the NSLDS II

1.3 Executive Summary

As of spring 2002, NSLDS contained comprehensive loan related information on over 45 million Title IV recipients, 143 million loans, 31,000 lenders, 18,000 schools and 60 guaranty agencies. Information stored inside NSLDS dates back to 1958.

NSLDS was originally intended to support transactional business functions such as student aid eligibility, default rates, and enrollment. The quantity of detailed data required to support the system functions has enabled NSLDS to evolve into a data repository.

The ultimate goal of NSLDS II will be to deliver reliable and flexible information that users need, on demand. The data warehouse is merely a means to achieve this end given the many ways that information for any given subject area can be stored throughout the enterprise.

Through a combination of tools already in use by FSA, as well as a new data warehouse product, NSLDS II will provide an efficient, cost effective solution that will not only improve ease of access but also significantly reduce operating expenses.

Data acquisition will employ a combination of the EAI Bus to guarantee delivery of batch files and Informatica PowerCenter for the staging and loading of data contained in batch files. Data storage will be handled by IBM DB2 EEE, which is specifically designed for data warehousing applications. Data access will be provided through MicroStrategy 7*i*.

1.4 Scope

The scope of the NSLDS II Technical Architecture Plan is to provide an overview of the components that comprise the NSLDS II technical architecture, as well as provide a summary of the technical architecture decisions.

1.5 Intended Audience

This document is intended to assist the client, detailed design team and the development team in understanding the technical framework in which the NSLDS II system will be constructed.

2 Legacy NSLDS Mainframe Technical Overview

The legacy NSLDS can be divided into three main components:

- Data Acquisition – The acquisition strategy for populating the data warehouse with data
- Data Storage – The data storage solution for the data warehouse
- Data Access – Provide the end users with reports and analytics of the data

2.1 Legacy NSLDS Mainframe Data Acquisition Overview

The legacy NSLDS data acquisition layer handles multiple data feeds, with extensive editing and transformation capabilities to ensure that data is suitable for loading into NSLDS. NSLDS currently uses COOL: Gen-generated COBOL running in an IBM mainframe environment to perform data acquisition.

The legacy NSLDS data acquisition layer is comprised of a number of batch processes that are scheduled to run nightly. Interface partners interact with NSLDS using magnetic tape or on electronic interface. Some interface partners interface electronically using the Student Aid Internet Gateway (SAIG), while others interface directly with NSLDS via FTP.

External users interfacing with NSLDS prepare their data submissions using the DataPrep software included as part of the EDConnect software package. Once data has been submitted to the NSLDS mainframe, COOL:Gen programs are used to check the integrity of the data in a staging area before the data is written to NSLDS.

Due to the fact that data acquisition is controlled through the use of the COOL:Gen generated COBOL code, changes to the mainframe data acquisition process are cumbersome and time consuming. The COBOL code must be compiled each time a change is made to the process. The slightest change to the load of an interface or a production environment requires a significant development and testing effort to validate that a requested change has been successfully implemented with no undesired repercussions.

2.2 Legacy NSLDS Mainframe Data Storage Overview

2.2.1 Technology and Architecture

The current NSLDS storage layer is an IBM DB2 database, running on IBM's OS/390 mainframe hosted at the Virtual Data Center (VDC) in Meridian, Connecticut.

The NSLDS logical and physical data models are modified using the custom development CASE tool Cool: Gen. Any changes made to NSLDS are made using the Cool: Gen tool. The DB2 database houses three separate development, test and production environments. COBOL programs and the bulk loading capabilities of DB2 are used to add or update data into NSLDS.

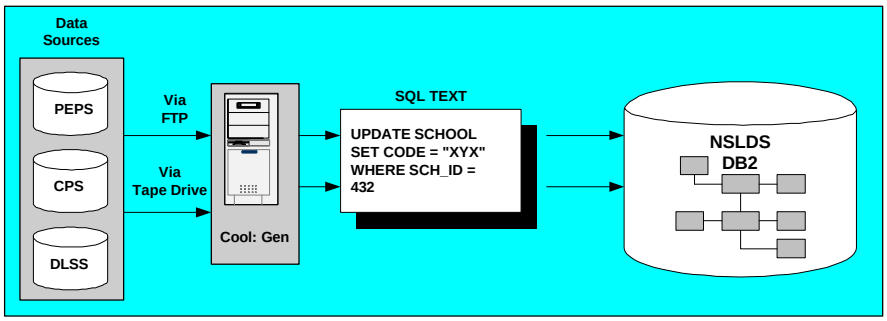


Figure 1, NSLDS Mainframe Data Acquisition and Storage Process

All queries are performed using the Structured Query Language, or SQL. This is done through embedded SQL in COBOL, the Query Management Facility (QMF) tool or IBM DB2's own stored procedure language. Queries include all user queries, data extracts, views and pre-aggregations of the data.

2.2.2 Database Design

The legacy NSLDS is a relational database and is optimized in third normal form. The database has been constructed to eliminate redundant data storage, which can reduce the performance of multi-join queries run against the database. The objective of a third normal form database is to isolate the data so that inserts, updates, and deletes can be made in just one table and then reflected throughout the rest of the database via the relationships between the tables. To compensate for performance issues inherent in this design, frequently asked information is derived from the main database and stored in views and aggregation tables. These tables have been created to support the required analytics.

The logical database design has been separated into seven subject areas, which can be examined to understand the basic structure of the system, the relationships between the objects, and logically pick out what is stored and where.

The following table lists the seven major subject areas of NSLDS:

Subject Area	Description
Students	Individuals whose education or partial education is funded through Title IV aid programs.
Schools	Institutions of higher education that participate in Title IV aid programs.
Lenders	Financial institutions that lend money under Title IV aid programs.
Guaranty Agencies	Agencies that guarantee Title IV loans for lenders. These agencies also aid in the administration of collections of defaulted loans, reinsurance of defaulted loans, and disbursement of Supplemental Preclaim Assistance for the FFEL program.
Loans	Money borrowed for funding a pre-approved educational

	program including disbursements, cancellations, deferments, refunds, loan aggregates, and uncollectables. These do not include grants.
FDLP Servicer	Organization that services FDLP loans for ED.
Default Rates	The creation of default rates for lenders, schools, and GAs related to the organizations' loans in the FFEL program.

Table 1, NSLDS Core Area Subject Model

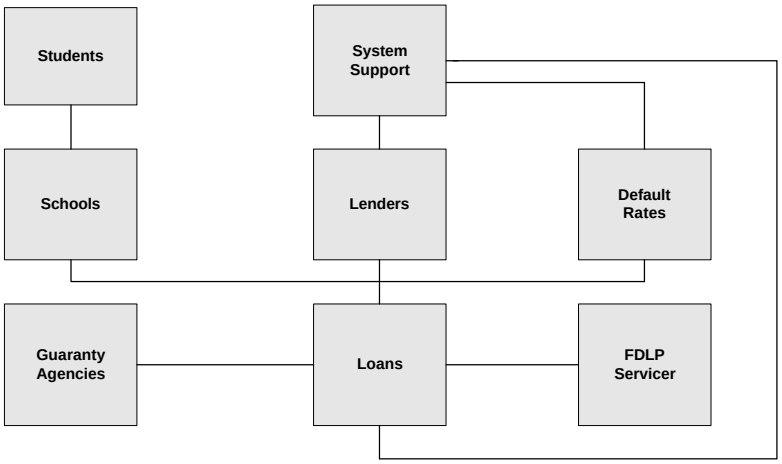


Figure 2, NSLDS Subject Areas

2.2.3 Database Considerations

NSLDS was developed as an Online Transaction Processing System (OLTP), which is designed to support the most current information needs of systems and users. NSLDS aids other transactional systems such as CPS, PEPS and DLSS, but also has grown to act as FSA's analytical Title IV data warehouse. As the amount of data has grown the current database design is no longer adequate enough to support all of FSA's analytical needs.

The database design prevents NSLDS from achieving the best results for its users for several reasons:

- IBM DB2 software running on a mainframe is not optimized for data warehouses
- NSLDS is setup as a transactional database, but is primarily used to perform analytic processing
- Application dependent environment

Based on IBM's project information, the mainframe version of DB2 currently used to house NSLDS is not optimized for handling multi-terabyte databases. Many other database tools are far superior in handling a multi-terabyte databases, and can deliver much faster query performance to users. Implementing a new database tool more capable of handling large amounts of data represents an opportunity for improvement.

The legacy NSLDS version of DB2 is unable to handle large complex group-based analytical queries directly from the main database. In order for NSLDS to answer these queries, views, aggregate tables and extracts must be created to allow users access to this information. Unless these views, aggregates and extracts are tightly maintained, data integrity and freshness becomes a problem. By using a dimensional database design and a database management system that is optimized for high-volume and high-performance queries, users will be better able to access that data.

Finally, NSLDS tightly interweaves application logic with database design, which affects the flexibility and scalability of the system. Since changes to the database design are dependent on the application logic, when a new requirement is added the database the application logic must also be updated.

2.3 Legacy NSLDS Mainframe Data Access Overview

To access legacy NSLDS, users at Department of Education (ED) and regional offices must have access to the Department of Education's Local Area Network (EDNet LAN).

GA and school users access NSLDS through SAIG. This network connects users with ED and its partners and customers, allowing them to communicate directly with the NSLDS database. Users who sign up for a SAIG account are assigned a "mailbox" that is used both to request and to receive data from NSLDS. GA and school users access NSLDS online functions through NET*CONNECT and batch files through EDConnect.

ED users access NSLDS through a mainframe terminal emulator that enables the creation a TSO, or Time Sharing Option session. This provides access to the Query Management Facility (QMF), Data File Download function, Report Management and Distribution System (RMDS), and a variety of other advanced functions.

- Data File Download allows users to download a report or extract from the mainframe to their PC.
- QMF allows users to query the NSLDS databases on an ad hoc basis. QMF queries can be created, saved, and executed on the NSLDS database.
- RMDS allows users to browse reports online, share them online with other authorized users, print all or selected pages of a report, or route all or selected pages of a report to diskette, magnetic tape, a SAIG mailbox, or a data set for download to a PC.

QMF is used to enter requests to retrieve data from NSLDS. QMF allows users to access NSLDS data through three types of queries:

- Structured Query Language (SQL) – SQL is a language used for writing queries that allows for the retrieval, creation, and maintenance of data located in a database.
- Prompted Queries – A set of prompts is used to gather information from the user to generate a query.

- Query By Example (QBE) – QBE is method of developing queries using very few keystrokes.

Data can be accessed in NSLDS using any of the three methods listed above; although, the most common technique for accessing data is using SQL. QMF does not contain data, instead it provides a place for legacy NSLDS users to enter queries and submit them to DB2 for processing. Once DB2 has processed the query, the results are returned to QMF.

QMF is comprised of five areas or object types:

- Profile—The QMF user profile, which includes details about the QMF environment.
- Query—The query that is being written or the most recent query that was run.
- Data—The data selected in the most recent query run.
- Form—The format that will be used to presented the data.
- PROC—The current procedure being written or the most recent procedure. A procedure is similar to macros in that it contains many commands that are run together.

Objects in the database storage areas are temporary; therefore, special actions must be taken to save the object. Legacy NSLDS data is located in DB2 on the mainframe. After the query is submitted, DB2 selects the data and returns it to QMF. QMF then combines the data with a format to create a report. Any changes to the query require the query to be resubmitted to DB2.

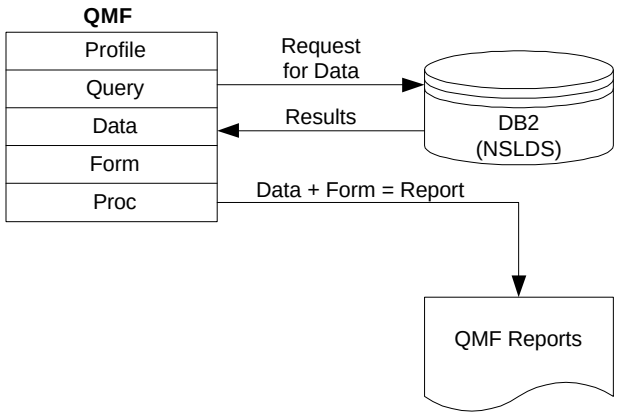


Figure 3, QMF Report Generation Process

When writing a QMF query, the user must have the following information:

- Table name
- Column names
- Row conditions
- The sequence in which the data should appear (for example, ascending by name)

A cost estimate of how hard the database manager will have to work to process the request is displayed when the query is run. These estimates are seldom precise. Queries that take too

long to run will be automatically interrupted or canceled. If the query exceeds the time limit or retrieves an excessive number of rows, processing may be interrupted as well.

The NSLDS data access architecture has several shortcomings including:

- Access to EDNet LAN and QMF is required to run ad hoc queries.
- An understanding of SQL is required to develop ad hoc queries in QMF.
- To drill down or reformat a query, the query must be resubmitted to DB2 for additional processing.
- Complex queries require long processing times and may be automatically interrupted or canceled by the system.

3 Data Warehouse Architecture

3.1 Architectural Overview

A data warehouse is a centralized database that collects, organizes and stores data from operational systems to provide a single source of integrated and historical data for the purpose of end-user reporting and analysis.

A data warehouse is defined by four main characteristics:

- Subject-oriented – NSLDS II will be organized around major subject areas such as loans, students, schools, and lenders.
- Integrated – A data warehouse provides the facility for integration in a heterogeneous, fragmented environment of independent application systems, where the data is stored in multiple, incompatible formats.
- Time-variant – The data warehouse organizes and stores the data needed for informational and analytical processing over an extended historical time range.
- Non-volatile – Changes to the data warehouse environment occur in a controlled and scheduled manner, unlike the more volatile OLTP environment in which updates continually occur.

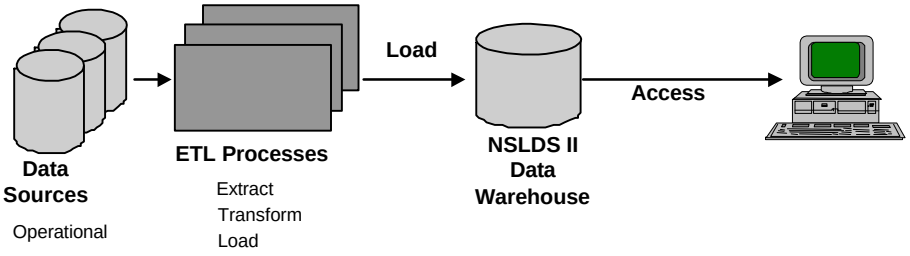


Figure 4, Simplified NSLDS II Data Warehouse Architecture

3.2 Data Warehouse Architecture Drivers

The data warehouse architecture design will be determined by where data is placed, how it is distributed, and the physical and logical design of the database. The five requirements that drive the NSLDS II architecture are as follows:

- Granularity of data
- Data retention and timeliness
- Reporting capability
- Availability
- Scalability

3.2.1 Granularity of Data

Data granularity refers to the degree of data aggregation or summarization. The lower the summarization of the data, the lower the level of granularity. Similarly, the higher the summarization, the higher the level of granularity. The different levels of granularity help the users of the warehouse access data quickly and accurately.

- Detail (Atomic, raw) = lowest possible granularity
- Lightly summarized = medium granularity
- Highly summarized = highest granularity

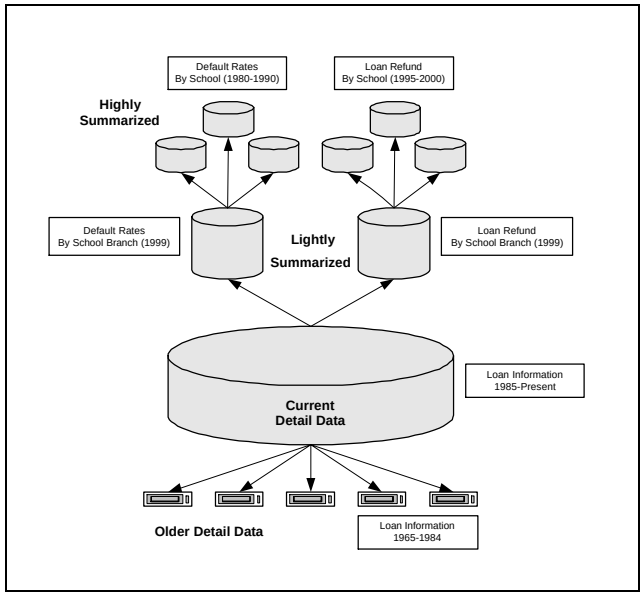


Figure 5, Data Structure Within the NSLDS II Data Warehouse

3.2.2 Data Retention and Timeliness

Data retention and timeliness refers to the data freshness factor. The user requirements should specify how current the data should be kept. These requirements should also address the issue of archived data and how long the data should be kept in the data warehouse before it is archived. These factors drive the replication and/or distribution of data warehouse data from the central site to reside closer to the user, or to the improvement in network bandwidth so that distributed users can perform reporting tasks from a central site.

3.2.3 Reporting Capability

Reporting access needs can range from simple reports and queries to detailed analysis and knowledge discovery.

3.2.4 Availability

Availability requirements are driven from the end-user perspective, such as whether they require 24x7 access to data and constant uptime.

3.2.5 Scalability

Scalability is the ability to increase the system load in terms of the number of users, data volume or processing without significantly decreasing response time or incurring costly hardware/software expenses.

4 Existing FSA Technical Framework Overview

The NSLDS II technical architecture will be integrated with the existing FSA technical architecture framework, which incorporates a number of established enterprise-wide solutions (as described in the Technology Standards and Products Guide v. 2.2) to assist in data acquisition and access including:

- EAI Bus
- Student Aid Internet Gateway (SAIG)
- Informatica
- MicroStrategy

4.1 EAI Bus

The EAI Bus incorporates a number of tools to meet its goal, including products from the IBM MQSeries family, as well as a plug in application to allow for file transfer. Additionally, to facilitate interfaces with the SAIG System, the EAI Core Technologies team has developed an adapter that retrieves files from SAIG and sends them across the EAI Bus, and places files sent to SAIG over the EAI Bus onto the SAIG Mailboxing system.

The architecture being used to transfer files between FSA internal systems has been repeatedly tested through a number of interfaces previously migrated successfully into production. As part of the Common Origination and Disbursement (COD) project, the EAI Core Team has established interfaces between the EAI Bus and the following systems that will interface with NSLDS II:

- COD
- CRM4FSA (Consistent Answers)
- CPS
- DLSS
- FMS
- Financial Partners DataMart
- Ombudsman Data System
- PEPS
- SAIG

DMCS (FFEL/DCS replacement) is also slated to have an interface with the EAI Bus as part of Release 3 of EAI Core Infrastructure, scheduled for completion prior to the implementation of NSLDS II.

These interfaces have successfully been tested to be successful in transferring large quantities of data reliably and efficiently. Should the volume of data produced by the interfaces between these systems and NSLDS II exceed current EAI Bus capacity, the EAI Bus provides a configurable, scaleable architecture that can be expanded to meet the additional traffic load generated by NSLDS II.

4.1.1 MQSeries

IBM MQSeries provides application-programming services that enable application programs to communicate with each other using messages and queues. This form of communication is referred to as commercial messaging. It provides assured, once-only delivery of messages. Using MQSeries means that you can separate application programs, so that the program sending a message can continue processing without having to wait for a reply from the receiver. If the receiver, or the communication channel to it, is temporarily unavailable, the message can be forwarded later. MQSeries also provides mechanisms for providing acknowledgements of messages received.

4.1.2 DataIntegrator (DI)

CommerceQuest DataIntegrator is a file transfer utility that leverages MQSeries to transport files across different information technology (IT) environments. The DataIntegrator provides the following functionality:

- Moves and accepts files among all supported platforms including, UNIX, OpenVMS, OS/390, and Windows NT
- Provides data compression if needed
- Performs binary to ASCII conversion
- Transfers files regardless of the size, format, or destination
- Allows individual status checking for any phase of the file transfer across the enterprise
- Allows customized data exits

DataIntegrator can break apart large files and send them as smaller MQ messages to a destination MQ queue and reassemble the MQ messages into a file at the destination. Pooling is available to load balance between parallel MQ channels when files exceed a specified limit. The maximum size of the MQ messages can be configured and any files exceeding that limit will be split up and sent across multiple MQ channels. This allows for load balancing and the most efficient utilization of network resources.

4.1.3 SAIG Get/Put Adapter

The SAIG Get/Put Adapter is a set of code that was developed by the EAI Core team. These two adapters help to link the mailbox infrastructure provided by SAIG to the messaging infrastructure provided by the EAI Bus. Since Data Integrator begins its send process by capturing a file, and ultimately delivers a file, a utility was required to input and output these files to the database structure in which SAIG operates.

While SAIG offers its users the EasyAccess Client for the purpose of interfacing with the SAIG server, this solution requires the user to download a compressed file and run the decompression agent locally. The Get and Put Adapters incorporate a connector API provided by bTrade (in order to interface with SAIG), a compression/decompression agent API (also provided by bTrade) and a DataIntegrator invocation in order to allow files to be sent to and from the EAI messaging layer.

The integration of SAIG to the FSA EAI architecture using the SAIG Get/Put Adapters allows systems to interface with SAIG in exactly the same manner that they do with every other EAI Bus-enabled system. Files enter the Bus infrastructure from SAIG in exactly the same state that they do from other systems with respect to compression. The EAI DataIntegrator product runs its own compression algorithm on the files it processes to enhance transfer efficiency.

4.2 Student Aid Internet Gateway (SAIG)

The Student Aid Internet Gateway (SAIG) Portal provides telecommunications support for the delivery and administration of Title IV programs via the Internet. This tool promotes electronic exchange of Title IV information between differing educational and other types of entities over the Internet.

SAIG Portal's core is composed of a HP-UX midrange that supports the mailboxing system and a NT/IIS server running a web administrative application. Supplementary systems include mainframe, enrollment web application, and PC systems. The SAIG Portal is a Commercial Off-the-Shelf (COTS) application from bTrade.com and is made up of the following core components:

- Secure Portal – An open architecture gateway that is used as the 'mailbox' application for the storing and retrieval of data.
- EasyAccess - The client software used to send/receive (FTP) SFA data transmissions securely over the Internet using SSL 3.0 and the Diffie-Hellman Dynamic Key Exchange algorithm.

SecureManager - Product that will be used to manage Title IV destination point mailboxes and data. This is referred to as TDCommunity Manager or Online Secure Manager (OSM).

4.3 Informatica

Informatica PowerCenter provides an environment that allows data to be loaded into a centralized location, such as a data mart, data warehouse, or operational data store (ODS). Data can be extracted from multiple sources, transformed according to business logic built in the client application, and loaded into file and relational targets. Informatica provides the following integrated components:

- **Informatica Repository –** Informatica repository is at the center of the Informatica suite. Users create a set of metadata tables within the repository database that the Informatica applications and tools access. The Informatica Client and Server access the repository to save and retrieve metadata.
- **Informatica Client –** Use the Informatica Client to manage users, define sources and targets, build mappings and applets with the transformation logic, and create sessions to run the mapping logic. The Informatica Client has three client applications: Repository Manager, Designer, and Server Manager.
- **Informatica Server –** Informatica Server extracts the source data, performs the data transformation and loads the transformed data into the targets. At runtime, Informatica

Server can also be configured to perform data integrity checks, and therefore serves as the implicit location of the data staging process.

4.4 MicroStrategy

MicroStrategy is an object oriented and web based tool that provides reporting, analysis, and information delivery services. MicroStrategy consists of products that perform the following functionality:

- Report serving
- Interactive reporting and analysis
- Design and administration
- Integration with other applications

5 Vendor Selection Overview

The NSLDS II Reengineering team evaluated the market for database tools optimized for a multi-terabyte data warehouse. When evaluating the database vendors, the following factors were taken into consideration:

- Overall affordability of the data warehouse package
- Long term cost reduction
- Vendor experience in implementing large data warehouses
- Ease of use, manageability, and configuration
- High degree of availability and scalability

The database vendors were evaluated against the following criteria:

- Credentials for Multi-terabyte Installations
  - Number of relevant installations
  - Satisfied client references
  - Independent Ratings from Gartner, Giga, Meta, etc.
- Key Technology Areas
  - Compatibility / affinity with existing FSA technology
  - Application (Siebel, Oracle, Financials)
  - Architecture (Informatica, MicroStrategy, MQ Series, WebSphere)
  - Hardware (Sun, HP, NT)
- Other Criteria
  - Fast data load performance with system availability during the data load
  - Fast query performance with fine control on query limits
  - Scalability from 4 terabytes to 10+ terabytes

5.1 Selection Process

The NSLDS Reengineering team went through a multi-step process in choosing the database vendor.

Step One – Identify Potential Vendors
The first step in the process was to identify those organizations that were in the business of implementing multi-terabyte databases. Five companies were identified. These five companies had enough credibility and market share to be considered as possible candidates:

- Teradata
- Oracle
- Microsoft
- Sybase
- IBM

Step Two – Internal Research
Internal research was conducted on each of the five vendors. Data from independent research firms including Gartner, Giga, Meta and Forrester was collected and reviewed to determine how each vendor was rated in the data warehousing market space. The NSLDS II Reengineering team also conducted Accenture internal research. The documents found on this medium primarily focused on existing documentation from projects that had chosen these vendors for their data warehousing projects. The team was able to gain valuable information on each of the potential vendors through this process.

Internal research provided enough support for the elimination of Sybase due to the lack of evidence that Sybase had successfully implemented a multi-terabyte database for a client. Additionally, Sybase's overall market share excluded them from further consideration.

Step Three – Vendor Information Meetings
After the initial research was conducted the NSLDS II Reengineering team held an information gathering session with each of the remaining vendors. Each vendor was asked to provide examples of multi-terabyte installations including the number of relevant installations, satisfied client references and independent ratings. Vendors were also asked to discuss the initial and ongoing total cost of ownership.

By using the information gathered through internal research and the vendor meetings it was determined that Microsoft and Oracle did not have enough relevant multi-terabyte databases installations to be considered. Microsoft was unable to provide a single example of a multi-terabyte installation and Oracle provided only one example of a 32 terabyte test database that they had configured.

Step Four – Conference Calls with Satisfied Customers
Conference calls where then conducted with satisfied customers who had either installed DB2 EEE or Teradata. Teradata clients with successful implementations of multi-terabyte data warehouses were interviewed. These clients included the Bank of America, Sprint, and USPS. Each client spoke of the solid performance of Teradata and the low maintenance costs of the hardware and software.

IBM clients with successful implementations of multi-terabyte data warehouses included Discover Financial Services, Freddie Mac, Cabala, and Visa EU. These clients stated their satisfaction in choosing IBM and discussed how IBM fit with their existing technical architecture, use of open standards and partnerial relationships.

Step Five – Submit Request for Proposal
A request for proposal was submitted to IBM and Teradata. In this document the vendors were asked to provide information on how they would provide data warehouse capabilities based upon the guidelines issued in the RFP.

The guidelines included detailed information on the current database server with the number of NSLDS users. It also detailed the data sizing and database design. The new data warehouse

provided by either company needed to be able to grow 50-100% each year, growing to 20 terabytes by the year 2007. Other details in the proposal included tools that support the target architecture, types of queries and reports used, sources of data, availability and disaster recovery.

Step Six – Database Vendor Decision
After evaluating the responses to the RFP and the other information gathered during the vendor selection, it was determined that the IBM DB2 EEE database solution would best meet the requirements of NSLDS II Reengineering project.

5.2 Scoring Summary

IBM was chosen as the preferred solution for the NSLDS Reengineering platform for several reasons including credentials, cost, research, and technology.

IBM received high ratings from all the independent evaluators such as Gartner, Giga and Meta, which stated that IBM has consistently had one of the top data warehousing tools on the market. Additionally, IBM was able to demonstrate many multi-terabyte data warehouse installations and proved that DB2 EEE is more than capable of handling a large data warehouse.

IBM demonstrated certain technical advantages including scalability through its "shared nothing" architecture, which works well on both IBM and non-IBM platforms. DB2 software and hardware can also be upgraded in a cost effective way with little or no disruption to the system. DB2 is compatible with Informatica, MQ Series, and MicroStrategy, which are all FSA standards.

Another important factor in choosing IBM was future developments. While NCR is currently the market leader in larger multi-terabyte data warehouses, IBM is continuing to make significant advancements and is positioning itself to become the benchmark for data warehousing within the next few years. IBM has already invested a substantial amount of resources toward the research, development and rapid deployment of new features. In contrast, NCR is focusing its attention on moving Teradata onto NT platforms and making smaller changes to its RDBMS engine. By choosing IBM as the database vendor, NSLDS II will be well positioned for future growth in the years to come.

6 NSLDS II System-Specific Data Acquisition Approach

NSLDS II will have interfaces with a number of systems and external parties that can be logically grouped into the following categories:

- Legacy NSLDS During Data Migration Stage
- External Systems Interfacing via SAIG
- External Systems Interfacing via Magnetic Tape
- Internal FSA Systems Interfacing via the EAI Bus
- Internal FSA Systems Interfacing via Magnetic Tape (temporary)

Each type of interface listed above has its own nuances and challenges that necessitate a different data acquisition plan, with many similarities, for each category. These categories group the interfaces along both logically and technically similar lines. This document provides an overview of the current solution for each of these groups, and elaborates on the proposed solution for the NSLDS II system.

6.1 Legacy NSLDS During Data Migration Stage

During the Data Migration, a large quantity of data residing in the legacy NSLDS will be transported to a new instance of the NSLDS active database consisting of core tables, as well as the STAB database. The shell for this data will be generated prior to any conversion.

Informatica PowerConnect for mainframe will be used to transfer of data from legacy NSLDS to the Informatica Server which will perform the transformations necessary to populate the target NSLDS II database.

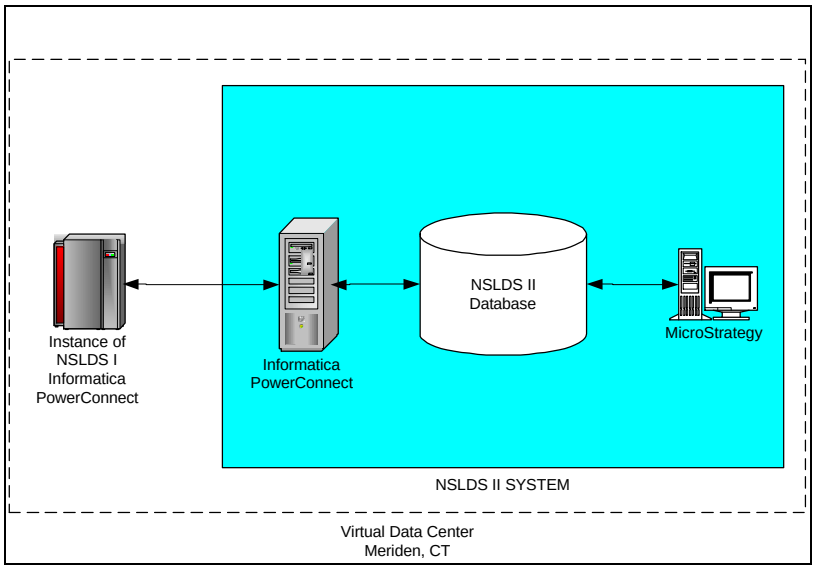


Figure 6, NSLDS to NSLDS II Data Migration

6.2 External Systems Interfacing Via SAIG

There are a number of external systems that will interface with NSLDS II via the SAIG. Among these systems are schools that report enrollment information, as well as the Clearinghouse (National Student Clearinghouse) and Guaranty Agencies. Since these external systems' relationship with SAIG has been established through the deployment of the SAIG system, the backend process of retrieving files from and sending files to the NSLDS II mailbox at SAIG will simply be re-pointed from the Legacy NSLDS system to the reengineered NSLDS II.

To meet the goal of transferring data files between SAIG and the reengineered NSLDS II, the NSLDS team will take advantage of the connectivity already established by the EAI Core Team. Data files waiting at the SAIG server will be accessed using the SAIG MQSeries Get Adapter. This adapter enables access to the file so that the product can transmit it to any EAI Bus customer, in this case NSLDS II, via DataIntegrator and MQSeries. Data Integrator will deliver the file to a specified location on the NSLDS II system and trigger a job to process the file through Informatica. During the Informatica process, data contained in the original files will be entered into the NSLDS II database.

For files that will travel in the opposite direction, from NSLDS II to the external system via SAIG, the same process will work in exactly the opposite direction. Informatica will extract the appropriate data from the NSLDS II database via Informatica scripts, creating a batch file in the appropriate layout. Once the file creation step is complete, DataIntegrator will deliver the file to the SAIG Server and, upon successful delivery, trigger the SAIG MQSeries Put Adapter. SAIG will then deliver the file to the designated recipient where the current process for the external system to retrieve the file will be followed.

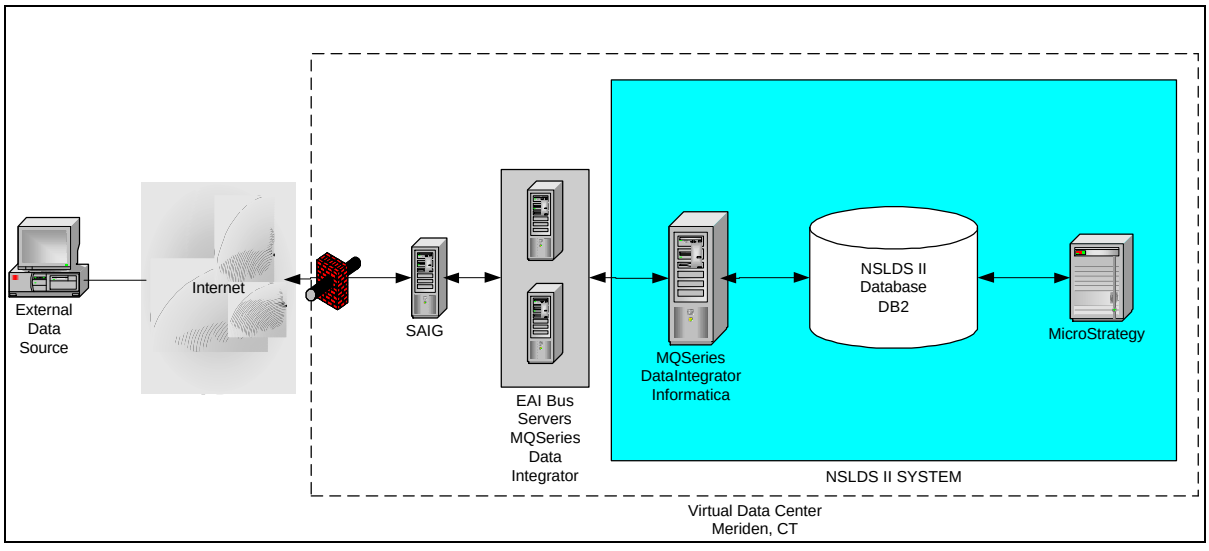


Figure 7, NSLDS II External Interfaces via SAIG

6.3 External Systems Interfacing via Tape-Stored Loads

Some of the external systems interface with the NSLDS II via magnetic tape. In situations where NSLDS II receives a file from an external source on magnetic tape, a tape drive connected to the NSLDS II Informatica server will load the file onto the Informatica Server. An Informatica script will then process the file to create database load language that can be executed against the NSLDS II database. In the opposite direction, an Informatica script will create a flat file that will then be saved to magnetic tape and sent back to the trading partner.

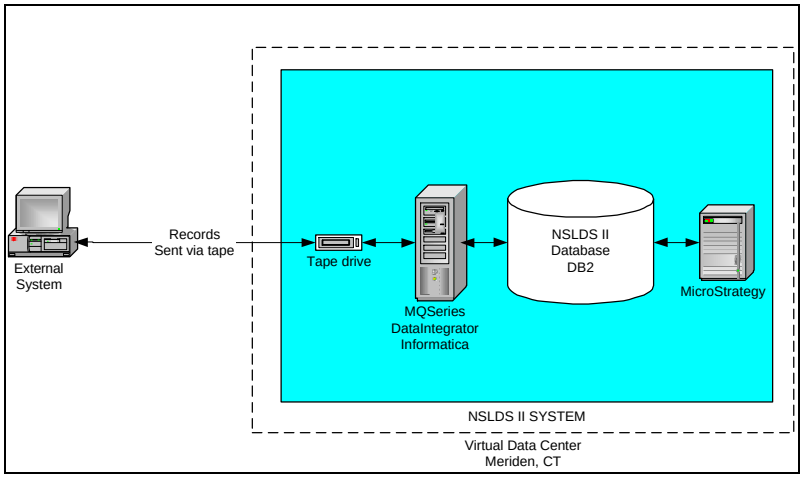


Figure 8, NSLDS II External Interface via Tape Drive

6.4 Internal FSA Systems Interfacing via the EAI Bus

To meet the goal of transferring data files between Internal FSA Systems and the reengineered NSLDS II system, the NSLDS team will once again take advantage of the EAI Bus connectivity already established by the EAI Core Team. Data files prepared by interfacing systems will be sent to NSLDS II via DataIntegrator and MQSeries. DataIntegrator will deliver the file to a specified location on the NSLDS II system and trigger a job to process the file with Informatica. Through the Informatica process, data contained in the original files will be entered into the NSLDS II database.

For files that will travel in the opposite direction, from NSLDS II to the internal FSA system, the same process will work in exactly the opposite direction. Informatica will extract the appropriate data from the NSLDS II database via Informatica scripts and create a batch file in the appropriate layout. Once the file creation step is complete, DataIntegrator will deliver the file to the interface partner and, upon successful delivery, the receiving system will begin to process the file in the same manner as with the legacy traditional FTP transfer process.

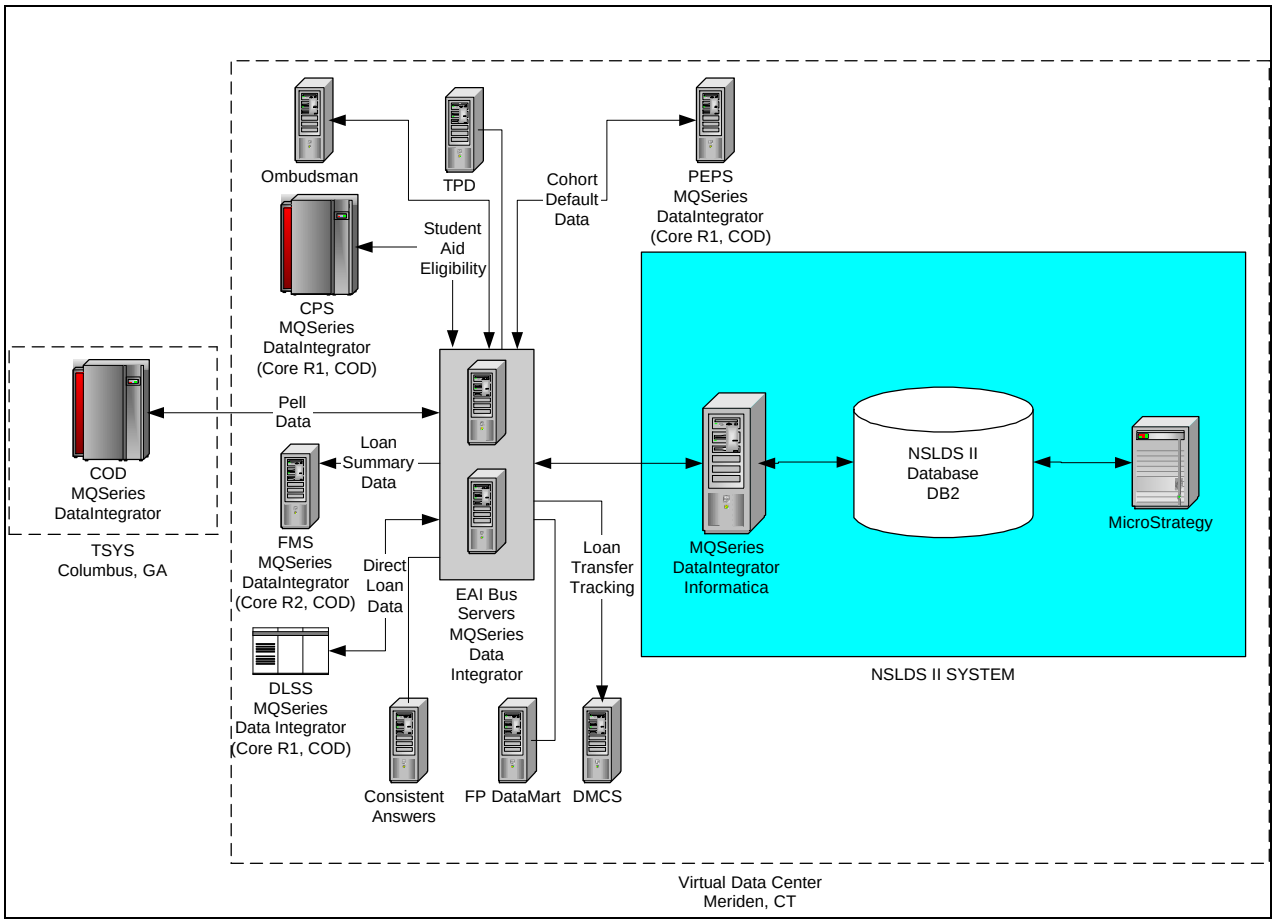


Figure 9, NSLDS II Internal Interface Environment

6.5 Internal FSA Systems Interfacing via Tape-Stored Loads (may be removeable)

The DMCS System currently interfaces with the NSLDS II system via magnetic tape. In the event that the DMCS reengineering is not complete as of the implementation of NSLDS II Release 1, the NSLDS II system will need to operate this interface via magnetic tape. Once the DMCS reengineering is complete, this interface will be moved to the EAI Bus and follows the description detailed in the previous section. **(Section 6.4 Internal FSA Systems Interfacing via the EAI Bus)** If this is the case, a tape drive, connected to the NSLDS II Informatica server will load the file onto the Informatica Server. An Informatica script will then process the file to create database load language that can be executed against the NSLDS II database. In the opposite direction, an Informatica script will create a flat file that will then be saved to magnetic tape and sent back to the trading partner.

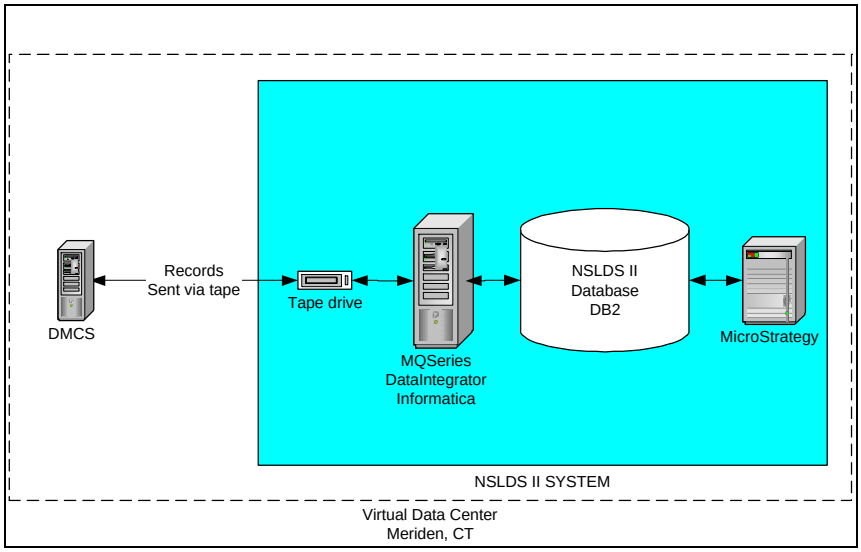


Figure 10, Possible DMCS Interface via Tape

7 NSLDS II Data Storage Approach

7.1 Data Storage Architecture

NSLDS II data will be stored in a DB2 EEE data warehouse running on a clustered IBM pSeries server, with FAStT500 Storage Arrays. This clustered architecture is extremely modular and will enable NSLDS II to grow in a very granular manner to meet the planned and future data warehousing needs of FSA.

DB2 EEE features a shared nothing database architecture that is extremely scalable. The proposed pSeries cluster provides corresponding shared nothing hardware architecture, which is uniquely suited to exploit the scalability advantages of DB2 EEE. DB2 EEE not only exploits the parallelism provided by the multiple servers comprising the cluster, but it also takes advantage of the parallelism provided by the multiple processors within each symmetric multi-processor (SMP) server. More specifically, DB2 EEE enables SMP parallelism within a database partition to facilitate better load balancing of the query workload. DB2 EEE provides full parallel execution, blending the best of both SMP and massively parallel processor (MPP) cluster architectures. Even database recovery is done in parallel.

When environment expansion is required to meet the ever-increasing data volumes of NSLDS II, each successive upgrade to the hardware infrastructure will provide a full complement of additional resources to ensure that scaling is nearly linear. Processing capacity, memory, I/O bandwidth, as well as bandwidth provided by the high-speed switch interconnect, all scale as the system grows. This modular addition of all components critical to the system's ability to scale, coupled with DB2 EEE's scalable design, is the reason for the overall extreme scalability of the proposed DB2 EEE data warehouse. In addition, DB2 EEE seamlessly integrates new hardware components into the database. Additional resources can be added on the fly, while the application is up and running. This modular growth capability, adding sets of processing power within an upgrade, will provide a predictable and consistent increase in capacity, easing the burden of capacity planning FSA.

This basic architecture consists of a central data warehouse server designed to meet the capacity and workload requirements. This architecture also supports the integration of MicroStrategy NT front-end servers for end-user access to the central data warehouse, as well as Informatica servers to support the necessary ETL processing.

Due to the scaling of DB2 EEE, data marts are not needed as a means to improve performance. Subject matter specific tables can be incorporated into the core warehouse, providing a "virtual data mart" within the warehouse. The DB2 EEE capabilities that enable this are:

- Advanced optimization, including views
- Automatic summary tables, which require no changes to user queries for exploitation and which are automatically maintained by DB2
- Support for relational OLAP SQL, such as Rollup and SQL statistics

- Advanced indexing, supporting index only access, bi-directional indices, dynamic bitmap index, and generated columns

7.2 Dimensional Modeling

Users of NSLDS II have placed more emphasis on accessing views and reports that read large amounts of detailed and summarized data. Due to the current database design and amount of data in the system this has become time consuming and costly. Dimensional modeling will allow NSLDS II to reach its goal of becoming a fully functioning data warehouse that enables its users and supporting FSA systems with the answers they need when they need them.

The reasons that NSLDS II will benefit from dimensional modeling include:

- Improved query performance
- Summarized Data Structures for reporting
- Predefined user access to data via fact tables
- Query repeatability via fact tables
- Dimensional views share integrated base structures

The objective of dimensional modeling is to represent FSA with a tool that is intuitive and provides high access performance. Dimensional modeling allows users to run queries and reports at both a more summarized and detailed level. Dimensional modeling enables trending and data analysis.

7.3 Dimensional Modeling Example

Through dimensional modeling, not only will users be able get faster results for existing reports and extracts, but they will also be able to ask questions that previously could not be asked. The current data model is currently not constructed to accommodate this kind of analysis, and often prevents certain types of queries and reports from being asked or created. Dimensional models, like the one proposed, are better geared for the type of analytics and ad-hoc queries run against large volume databases.

Consider this possible ad-hoc query that a user might execute in NSLDS II. Through dimensional modeling this type of questions can be easily answered in a timely manner.

"Display ABC College's total defaulted loan balance for the last 4 years."

Year	Defaulted Loan Balance
2001	$100,000
2000	$110,000
1999	$90,000
1998	$80,000

Table 2, Dimensional Modeling Example 1

Given these results, the NSLDS II user may want to "drill down" further into these results to see specific loan balance information for each year, perhaps asking to see the specific lender names that comprised the original result set.

Year	Lender Name	Defaulted Loan Balance
2001	City Bank	$40,000
	USA Bank	$25,000
	ACME Lenders	$35,000
2000	Federal Lending	$50,000
	ACME Lenders	$25,000
	University Bank	$35,000
1999	ACME Lenders	$30,000
	USA Bank	$30,000
	Federal Lenders	$30,000
1998	USA Bank	$40,000
	ACME Lenders	$20,000
	ELA Lending	$20,000

Table 3, Dimensional Modeling Example 2

Obtaining these more granular results would be extremely resource intensive and cumbersome if executed in the current system. However, using dimensional modeling the preceding result set could be filtered down into even more detail. For example, users would be able to access the particular name of the students that defaulted on their loans for the year 2001. This is a perfect example of how dimensional modeling will allow more detailed analysis for users.

Year	Lender	Student Name	Defaulted Loan Balance
2001	City Bank	John Doe	$20,000
		Mike Smith	$15,000
		Jane Doe	$5,000
	USA Bank	Steven Burke	$7,000
		Nathan Jones	$9,000
		Lewis Williams	$9,000
	ACME Lenders	Max Owens	$20,000
		Jerry Robinson	$15,000

Table 4, Dimensional Modeling Example 3

As illustrated in the examples above dimensional modeling empowers users with access to high-level summaries and enables them to further "drill down" into these results for even more detail. The current legacy NSLDS data model, or normalized model, is simply not built for analytics of this nature. To service these functions, the current NSLDS system supports the creation of aggregate tables to store these and other summaries. However, these aggregates consume more space and are more cumbersome to maintain given the nature of the normalized model. For these reasons, and many others, changing the database design to a dimensional model that uses a "star" or "snowflake" model is imperative to achieve the desired improvements for NSLDS II.

7.4 Current NSLDS Data Model

The current data model was built in the Information Engineering Facility (IEF) CASE tool, more recently known as COOL:Gen. A guiding principle of the IEF designing philosophy is that all processes can be grouped into like subject areas. Further, these subject areas can be used to group related entities.

Entities are defined as collections of business data on which business functions operate. Within this model, connections between these entities are called relationships. Entities may be related in one to one, one to many, zero to many or many to many fashion.

The current NSLDS data model is a 'loan-centric' database, meaning that the loan subject area is the central focus of the design. Therefore, most entities from the other subject areas, like Students, Schools and Lenders, are directly and/or indirectly related to an instance of a loan. As such, the LOAN table contains a large amount of information and has several tables that are related to it, each containing additional loan information. Since this data model is 'loan-centric', it is very inefficient to run queries that need to access data from multiple subject areas regarding non-loan related information. In this case, each query must "go through" the LOAN table, or use the LOAN table to create the necessary joins, to get the desired results. Given that the LOAN table contains roughly 140 millions rows, this is an expensive operation in terms of CPU time and user time. The following diagram illustrates this 'loan-centric' aspect of the current NSLDS data model structure.

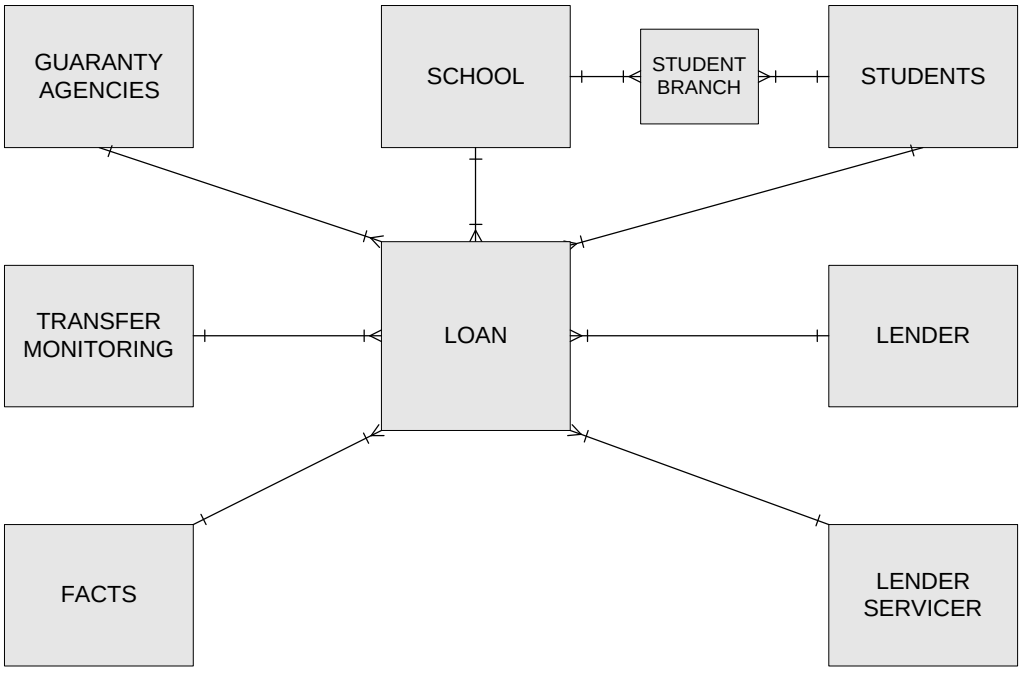


Figure 11, Current NSLDS Data Model

7.4.1 Evolution to new Logical Model

The current data model is used to support many reports. Reference the NSLDS Report Content Descriptions and System Procedures document for a complete list of reports targeted for implementation in NSLDS II. The purpose of this is to gain an understanding of just what fields and tables are used to generate current NSLDS reports. This is the first step in determining which tables are absolutely necessary to transfer into the new data warehouse.

After determining the reporting requirements, the next step is to analyze the current table structure. This will provide a means of determining which tables and columns map to current reports and thus need to be converted. Reference the NSLDS Data Conversion and Migration Strategy document for a complete list of tables targeted for conversion.

7.5 Target Logical Data Model

The new NSLDS II data model is designed as a dimensional logical model. A logical model lists database entities and their relationships with each other. The physical model, which will be designed as part of the next phase of work, will show how the entities relate to the physical structure of the database.

This section will describe each aspect of the new design by explaining each dimension and its role in the logical model. It is important to note that this design is only preliminary and does not entirely represent the final physical model.

The design of the new model adheres to the following requirements:

- Includes only information that is needed for querying and reporting
- Provides a structure that will be most efficient when used in conjunction with the MicroStrategy platform
- Provides the ability to drill both down and across dimensions efficiently
- Addresses the need for payment comparisons in relationship to time as well as looking at a “snapshot” of loan status
- Balance current reporting needs with potential new reports (reports not currently created in legacy NSLDS)

7.5.1 Preliminary Logical Model

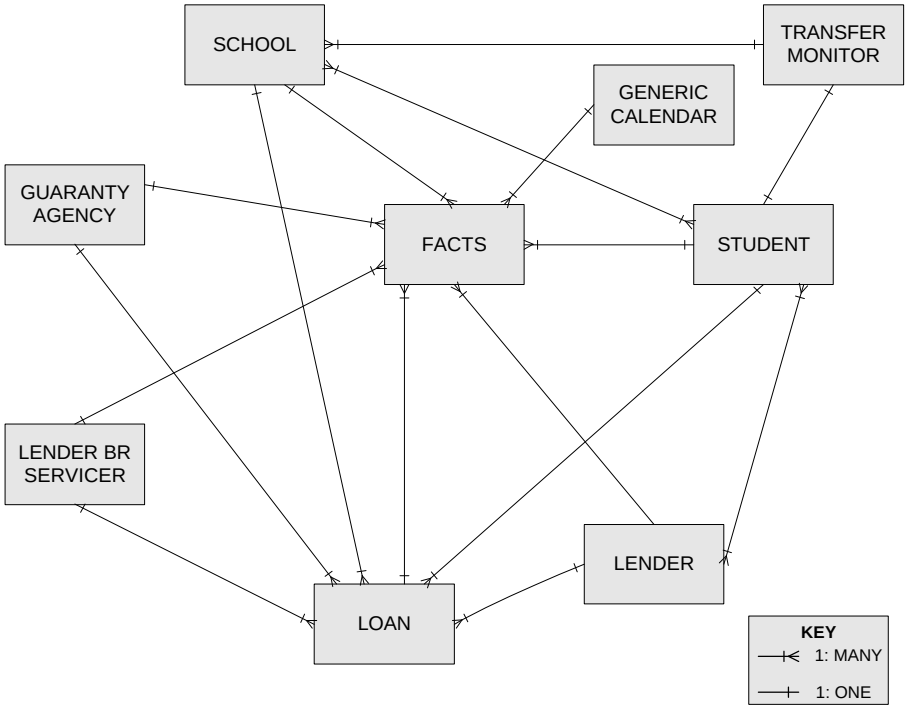


Figure 12, NSLDS II High Level Data Model

The logical model as shown above illustrates the relationships between the core dimensional entities. Each rectangle represents a single logical dimension. Dimensions are logical groupings of attributes. In the eventual physical model, these attributes will represent columns. These attributes will be the key focus areas for the physical structure. Each line between dimensions represents a relationship of the type one-to-one (1:1), one-to-many (1:M or M:1), or many-to-many (M:M). Although loan is still a key point in the model, notice that it is no longer the central point.

All dimensions are represented, in some manner, on the fact tables. Facts represent business metrics and information that can be aggregated. Some dimensions have an additional relationship between themselves allowing an analyst to more quickly drill between information, i.e. from the student level to the school level. Each dimension can also contain characteristic information related to the main focus of the dimension. For example, the school dimension contains all relevant school characteristics, such as address, branch, and type.

The following sections present all of the objects and their roles currently included in the data model. Please see Appendix A for a graphical display of intra-dimensional relationships.

7.5.2 Generic Calendar

Many reports show information over a time period. Time can be described not only by dates, but also by years, quarters and months. This relationship permits the analysis of data to be seen at one time level and quickly aggregated or drilled into another time level.

Attributes	Description
Date	Calendar dates
Month	Month of Year
Quarter	Quarter of Year
Year	Calendar year

The generic calendar dimension can also define time relationships for any number and type of calendars. This section will represent both Fiscal and Gregorian calendars.

7.5.3 Guaranty Agency

The Guaranty Agency dimension contains only the Guaranty Agency attribute. There are no other characteristics.

Attributes	Description
Guaranty Agency	An agency that acts as the primary insurer under the Federal Family Education Loan Program. GAs may be state-owned or private, non-profit institutions. They have a broad authority to establish policies and procedures for administering the FFEL Programs in accordance with the enabling statues and regulations. There are approximately 50.

7.5.4 Lender

The lender dimension contains characteristic information pertaining to the lender as well as original lender.

Attributes	Description
Lender	A private institution that provides capital for FFEL Programs. This may include credit unions, commercial banks, savings and loan associations, eligible schools, insurance companies, a single agency of a state, or pension funds. There are approximately 11,000 lender participants.
Lender Address	The street address of an institution issuing an FFEL Program Loan.
Original Lender	Six-digit ED code indicating the original lender of an FFEL Program promissory note.

7.5.5 Lender Branch Servicer

Lender Branch Servicer, while encapsulated within the lender in the current database, will be broken out into its own dimension in the new model.

Attributes	Description
Lender Branch Servicer	An organization/business that provides customer service support for all or a subset of borrowers associated with an FFEL Program lender.

7.5.6 Loan

The loan dimension contains many characteristic attributes. Each represents a certain aspect of the loan required for reporting.

Attributes	Description
Academic Level	Student's academic or grade-level at the school at the time the loan was guaranteed or disbursed.
Academic Yr	The academic year covered by the loan.
Cancellation Date	An added attribute indicating the latest cancellation date for a loan if that loan has been canceled.
Cancellation Type	One of the legal reasons for repayment of loan principal and/or interest being deferred (e.g., CP–Peace Corps, EH–Economic Hardship).
Class Begin Date	Date when classes are to begin, as certified by the school on the aid application, for the specific period covered by aid (e.g., loan, grant, CWS).
Class End Date	Date when classes are to end, as certified by the school on the aid application, for the specific period covered by aid.
Default Date	Date on which the loan defaulted according to the definition of a default in cohort default rate processing.
Default Region	The ED region responsible for a loan that has defaulted and is now retained by ED.
Deferment Start Date	An attribute indicating the latest deferment start date if a loan is in deferment.
Deferment Stop Date	An attribute indicating the latest deferment stop date if a loan is in deferment.
Deferment Type	An attribute indicating the latest deferment type if a loan is in deferment.
Disbursement Date	An attribute indicating the latest disbursement date for a loan.
Establish Date	The date the loan was established.
External GA ID	This attribute provides an alternate loan id, specific to the Guaranty Agency.
FFEL Duplicate ID	This attribute will uniquely identify those FFEL loans where Date of Guaranty does not provide sufficient uniqueness (i.e., for the same student at the same school for the same loan type, two or more FFEL loans may be issued on the same day). Values are A through Z.
Incorrect Loan Status	Code representing previous status of a student's loan, as determined by the loan's current holder.
Incorrect Loan Status Date	An added date to indicate when an occurrence of this entity is no longer valid.

Interest Rate	For an FFEL Program Loan, this is the interest rate at the time the guaranty is made. For Perkins Loans, this is the interest rate when the loan is disbursed.
Interest Rate Code	A code indicating whether the interest rate is variable (V) or fixed (F) for the life of a loan. (May also be 8 for 8/10).
Lender of Last Resort	Indicator that a loan is a lender-of-last-resort loan. Values are Y or blank.
Lender Branch Resp Begin Dt	Begin date when the lender was responsible for a particular loan
Lender Branch Resp End Dt	End date when the lender was responsible for a particular loan. If the value is 9999-12-31, the lender is still responsible.
Lender Branch Servicer Resp Begin Dt	Begin date when a Lender Branch servicer was responsible for servicing a particular loan for a particular lender.
Lender Branch Servicer Resp End Dt	End date when a Lender Branch servicer was responsible for servicing a particular loan for a particular Lender.
Loan	Long-term, low-interest capital obtained primarily through private lenders for the purpose of financing a student's educational expenses.
Loan Contact Type	A value indicating the type of organization to contact concerning an applicant's loan. Values are: GA–GA SCH–school EDR–ED region DLS–direct loan servicer SCS–school servicer LEN–lender LNS–lender servicer N/A - none
Loan Status	An attribute indicating the latest status of a loan.
Loan Status Date	An attribute indicating the latest status date of a loan.
Loan Type	An indicator for a specific loan program (e.g., CL–Consolidated, FI–Federally Insured Student Loan).
Loan Type Group	A code representing the full name of a loan program.
Maturity Date	Date a loan enters repayment or is to enter repayment status, regardless of whether or not the borrower actually begins making payments on that date.
Request Notification Date	The date a notification was sent by NSLDS to the school indicating that a lender has requested PCA/SPA assistance.
Request Type	Code indicating whether PCA or SPA has been requested on a loan.
Subsidized	An indicator of whether a Stafford loan is eligible for interest subsidy from ED (subsidized vs. non-subsidized).
Unsubsidized Flag	A flag indicating whether the applicant's loan has additional unsubsidized amount. Values are P–PLUS, H–HEAL, B–Both, or N–Neither.

7.5.7 Miscellaneous

Several attributes have a joint-relationship between more than one dimension. For that reason, those attributes are listed in this section. Additionally, many attributes are not directly related

to dimensions and are only important at the fact or atomic level. These attributes, known as degenerate attributes, are also listed here.

Attributes	Description
Adjusted Date	Date on which loan was adjusted to account for the unpaid refund.
Attending School	Six-digit ED code for uniquely identifying a SCHOOL and the two-digit ED code for uniquely identifying a BRANCH of a SCHOOL.
CDR Sub Type	Six-digit ED code for uniquely identifying a SCHOOL.
Claim File Date	The date for which a Guaranty Agency files a reinsurance claim with the government.
Claim Paid Date	The date on which the government pays a reinsurance claim to a Guaranty Agency.
Claim Payment Reason Cd	The reason a claim was paid to a GA.
COH Calculation Dt	The date on which the CDR was calculated.
COH Calculation Yr	The year for which the CDR is valid.
Collection Date	An attribute indicating the latest date for principal collections for a loan.
Date Claim in Sys	Date when this Insurance Claim Payment was added to NSLDS. This is needed since amounts may be transmitted to NSLDS at a date later than the actual event. Aggregate calculations will use this date to determine the month to which the amount applies.
Enrollment Cert Dt	Date as of which a school certified enrollment information for a student.
Enrollment Status	Code identifying Enrollment status of student. As of 11/15/2000, valid values are: 'A' (Approved Leave of Absence) 'D' (Deceased) 'F' (Full Time) 'G' (Graduated) 'H' (Half Time) 'L' (Less than Half Time) 'W' (Withdrawn) 'X' (Never Attended) 'Z' (No Record Found)
Enrollment Status Dt	Date as of which an Enrollment status is effective.
Error Code	The error code for this specific error.
Expected Grad Dt	The date on which a given student at a given school is expected to graduate or otherwise terminate a course of study.
Family Contribution Index	An index indicating the degree to which the family of an applicant for Title IV aid is expected to contribute to the cost of attending school. Generated through application processing.
Field in Error Code	The error code of field in error.
Forgiveness Date	Date on which loan forgiveness was applied.
Forgiveness Type	One of the legal reasons for forgiving all or part of a loan.

GA Actual Submittal Dt	Date (year, month, and day) the Extract file was created by the data provider.
FS Actual Submittal Dt	Date (year, month, and day) the Extract file was created by the data provider.
SCH Actual Submittal Dt	Date (year, month, and day) the Extract file was created by the data provider.
GA Applied Date	The date (year, month, and day) through which the GA applied IRS offset interest and/or principal collections to a loan.
GA Claim Date	The date for which a Guaranty Agency files a supplemental claim with the government.
GA Claim Reason Cd	The reason a claim was paid to a GA.
GA Collected Date	Date of principal and interest collected by GA for default collections and bankruptcy recovery.
GA Job Completion Dt	This attribute signifies the date that this submittal completed processing through NSLDS.
FS Job Completion Dt	This attribute signifies the date that this submittal completed processing through NSLDS.
SCH Job Completion Dt	This attribute signifies the date that this submittal completed processing through NSLDS.
GA Paid Date	The date on which a GA paid a default claim to a lender.
GA Payment Date	Month and year in which the Guaranty Agency received a supplemental preclaims assistance payment from ED.
GA Scheduled Submittal Dt	The Scheduled Submittal Date for this provider's submission.
FS Scheduled Submittal Dt	The Scheduled Submittal Date for this provider's submission.
SCH Scheduled Submittal Dt	The Scheduled Submittal Date for this provider's submission.
GA Received Dt	Date this submittal was received by NSLDS.
FS Received Dt	Date this submittal was received by NSLDS.
SCH Received Dt	Date this submittal was received by NSLDS.
Ins Claim Reason Cd	The reason a claim was paid to a lender including borrower death, disability bankruptcy, default, closed school or false certification.
Last Ins Cl Reason	An attribute indicating the latest insurance claim reason code for a loan.
Last Int Coll Date	An attribute indicating the latest date for an insurance claim payment for a loan.
Loan Payment Dt	Date that a payment was made toward a loan.
NSLDS User	The NSLDS user responsible for a task.
Out Int Bal Date	The date on which the current outstanding accrued interest balance status was updated or verified.
Out Prin Bal Date	Date on which the value in outstanding principal balance field was updated or verified.
Pell Award Year	The school year for which a Pell Grant is to be used to fund a student's education.
Pell Grant	Unique number assigned by the Pell system within a specific SAR record to identify a particular payment record.
Rehabilitation Ind	Yes/No Flag to indicate that a loan had been rehabilitated when it was repurchased. It is used to distinguish between Part D (Full Refund of Reinsurance Claims) and Part H (Rehabilitated Loans) on the 1189.
Rein Claim Refund Date	The date on which a refund was received by ED from a GA.

Reinsurance Rate Cd	Reinsurance rate at which the government paid a claim to the Guaranty Agency on a specific loan.
Repayment Plan Date	Date (month, day, year) on which the FDLP loan entered a particular repayment plan.
Repayment Plan Type	Code indicating the type of repayment plan for an FDLP loan in repayment. Permitted values: FF–fixed payment, fixed term; FE–fixed payment, extended term; GR–graduated repayment; IC–income contingent; and SP–special plan/secretary's option.
Reporting School	Six-digit ED code for uniquely identifying a SCHOOL and the two-digit ED code for uniquely identifying a BRANCH of a SCHOOL.
Repurchased Date	The date on which a previously defaulted loan is repurchased by a lender.
Rerun Flag	A flag indicating if the CDR has been recalculated.
School Pell Status	Status code reflecting the school's verification of the applicant data.
Source of Grad Dt	The organization code of the organization reporting the Anticipated Completion information.
Student SSN	The student's social security number in the Error Tracking Submittal File.
Submittal History Code	The GA code that is received on the Error Tracking Submittal File Header.
User Organization	Organization that the NSLDS user belongs to.

7.5.8 School

School and its related information are highlighted in this dimension. Historical information relating to school names is also available.

Attributes	Description
Action Code	A two-character code used in conjunction with Action Reason Code to determine a schools current state of eligibility according to PEPS.
Action Date	The date the status of the associated school branch changed.
Action Reason Code	This attribute is populated from the PEPS system. The OPE IDs data dictionary describes this 'reason code' as 'reason for submitting eligibility request'. In NSLDS the primary need for this data is for default calculation in finding if a school is closed and why.
Campus Based ID	An NSLDS generated code identifying the specific organization that has serviced a loan.
City	The city in which the branch of an educational institution resides.
Congressional District	The congressional district of the main institution.
Country	The country in which the branch of an educational institution resides. This will be blank for domestic schools.
Eligibility Status	The indicator specifying whether the institution is eligible to participate in TIV programs.

OPE ID	The school code and school branch code of the school/branch responsible for originating the loan. N/A is substituted when the OPE ID is one of the special IDs (88888800, 88888811, 99999900) for consolidated or refinanced loans or for unknown schools.
Operational Status	The current status of the school being Open, Closed or Merged.
Original School	Six-digit ED code and name which uniquely identifies the original school.
Original School Branch	Two-digit ED code and name which uniquely identifies the original branch of a school.
Region	The ED region in which the main institution is located.
School	The educational institution which a student attends and through which the student may request and receive Title IV program financial assistance.
School Branch	An educational institution representing a school or subdivisions of a school.
School Branch State	A two-digit alphabetic code identifying the state in which the branch of an educational institution resides.
School Certification	A flag indicating that an institution is eligible and certified to participate in Title IV programs. Valid values for this field are: Y–school or branch is eligible and certified; N–school or branch is either eligible, but not certified or not eligible and not certified; and blank–school or branch not in certification file.
School Ethnicity	Indicator of whether an institution has been classified as: 1. Native American college 2. Historically black college 3. Hispanic college 4. Traditionally black college 5. Ethnicity not reported
School State	The two digit alphabetic code identifying the state in which the school branch servicer resides.
School Type	Length of longest academic program in terms of academic hours or years.
SSCR Cycle	The month numbers selected by the school for participation in SSCR. SSCR is generated 6 times or cycles a year. Two cycles are mandatory; four are optional.
SSCR Required Indicator	Indicates whether SSCR Needs to be sent for the cycle. Values are Y for yes, to be generated and N or blank for not to be generated.

7.5.9 Student

The student dimension surrounds the student with various complementary information including historical addresses and social security numbers.

Attributes	Description

Aid Overpayment Disbursement Date	Disbursement Date that caused the overpayment.
Aid Type	There is no default value. The value may be PK for Perkins, SS for SSIG, SE for SEOG and PE for Pell.
Bankruptcy Status	A flag indicating the applicant has a loan with an active bankruptcy status.
Citizenship	Indicator of whether a student is or is not a U.S. citizen.
Current Name Indicator	Indicates which of the names for a student is the active or current name for that student. Values are Y or blank.
Current SSN	An attribute indicating current social security number for a Title IV aid recipient. This can be updated with corrections.
Defaulted Loan Status	A flag indicating the applicant has a loan with a defaulted status.
Demographic Request Year	Year when a student has demographic information requested from CPS.
Discharge Loan Status	A flag indicating the applicant has a loan with a discharged status.
Driver License Number	The number a student is issued by the state when receiving a license to drive.
Driver License State	The state which issues a student's driver's license.
Overpayment Entry Date	Date on which the overpayment was entered.
Overpayment Entry Username	NSLDS user id of user who updated the overpayments.
Overpayment Status	R indicates a repaid status. S indicates satisfactory arrangements made to repay loan. Y indicates an overpayment.
Overpayment Status Date	Date on which the overpayment was modified. Applicable when the overpayment changes from Y to R. When the overpayment is set to Y this date will be same as Create Date.
Plus Borrower	The individual who takes financial responsibility for the repayment of a student's PLUS Loan.
Plus Borrower State	The state code for an FFEL PLUS borrower state of residence at the time the loan was guaranteed.
Prescreen Run	The system date on which the prescreening run began which included the applicant.
Pseudo SSN Indicator	An attribute indicating the current single-element indicator showing whether the student's current Social Security Number is real or a pseudo.
Repaid Date	Date on which the overpayment was repaid. Applicable only when the overpayment indicator is equal to R.
Repayment Status	A flag indicating the applicant has a loan with a status indicating satisfactory repayment arrangements have been made.

Source	Contains the value for source of overpayment. It represents the owner of the overpayment, which is the school, the Department of Education, Debt Collection Systems. If it is held by the school, it will contain the value SCH. If held by the Department of Education it will contain the value EDR. When it is in transition, that is being moved from the school to the Department it will have the value TRF. The ED Debt Collection System can update the records that have the source value EDR or TRF. A school can update the record only when this value is SCH.
Student	A person attending an institution of higher education, and who has received or been the beneficiary of Title IV aid.
Student Address Type	The indicator for categorizing the address. P represents the permanent address for the student and A represents an alternative address for the student.
Student Full Address History	The addresses held by a student over the course of their aid history of the student. It will contain the permanent address for the student (P) and possible alternative addresses (A). The entity can track which addresses are current and from what source the address was supplied or confirmed.
Student Name History	A record of a student's name as it changes due to name changes or to errors. The same first and last name combination will only be stored once regardless of the number of times that combination appears as a changed name.
Student State	Represents a valid U.S. state code. Part of the permanent address for a student receiving an FFEL loan.
Transaction Number	The transaction number as provided by CPS. This is not used by NSLDS other than to pass it back to CPS without edit or modification.

7.5.10 Transfer Monitor

There are only two attributes related to transfer monitoring that were requested in reports. Both attributes also relate to the student.

Attributes	Description
Alert Date	The system timestamp when the alert was created.
DL MPN Flag	An A/C/I/N flag indicating the status of the student's Direct Loan Master Promissory Note. Values are Accepted–A, Closed–C, Inactive–I, none–N.
Monitoring Date	The date monitoring is to begin for the student at the school.

7.5.11 Facts

The following list of facts is defined at the basic level for all reporting needs. This list will not show percentages or year-to-date metrics that will be calculated from these base facts. Each fact has at least one entry-level, or dimension where the fact is valid. For example, the loan refund

table's refund amount is available at the Guaranty Agency, loan, student, school, lender, and lender servicer level and can thus be readily queried from any of those dimensions. On the other hand, the fact interest amount on the collection table is only defined with Guaranty Agency and loan as its entry-level. Hence, interest amount on that table cannot be found for a lender because it is not defined at that entry-level.

Facts are listed by table name for easier recognition.

Table Name	Fact Name	Entry Level
CDR Appeal	DUAL Denominator	COH Calculation Dt, Rerun Flag, Lender Branch, School Branch, Guaranty Agency
CDR Appeal	DUAL Numerator	
CDR Appeal	FDSLP Denominator	
CDR Appeal	FDSLP Numerator	
CDR Appeal	FFEL Denominator	
CDR Appeal	FFEL Numerator	
CDR Appeal	ICR Count	
CDR History	DIR Denominator	COH Calculation Dt, CDR Sub Type, Lender Branch, School Branch, Guaranty Agency
CDR History	DIR Dol in Def	
CDR History	DIR Dol in Rpmt	
CDR History	DIR Numerator	
CDR History	DUAL Denominator	
CDR History	DUAL Dol in Def	
CDR History	DUAL Dol in Rpmt	
CDR History	DUAL Numerator	
CDR History	FFEL Denominator	
CDR History	FFEL Dol in Def	
CDR History	FFEL Dol in Rpmt	
CDR History	FFEL Numerator	
CDR History	FFEL Total Claim Amt	
CDR History	ICR Count	
Collection	Cumulative Int Amt	GA Collected Date, Guaranty Agency, Loan
Collection	Cumulative Prn Amt	
Insurance Claim Payment	Amount Diff	GA Paid Date, Date Claim in Sys, Ins Claim Reason Cd, Guaranty Agency, Student, Lender, Loan
Insurance Claim Payment	Cumulative Amt	
Insurance Claim Refund	Cumulative Amt	
IRS Offset	Cumulative Int Col Amt	GA Applied Date, Guaranty Agency, Loan, Student
IRS Offset	Cumulative Prn Col Amt	
Loan Cancellation	Amount Diff	Cancellation Type, Cancellation Date, Guaranty Agency, Student, Lender, Loan
Loan Cancellation	Cumulative Amt	
Loan Disbursement	Amount Diff	Disbursement Date, Lender

Table Name	Fact Name	Entry Level
Loan Disbursement	Cumulative Amt	Branch, Student, Loan, School Branch, Guaranty Agency, Lender Branch Servicer
Loan Fact	Loan Amt	Out Prin Bal Date, Out Int Bal Date, Lender Branch, Loan, Guaranty Agency, Student, School Branch, Lender Branch Servicer
Loan Fact	Out Int Balance	
Loan Fact	Out Prin Balance	
Loan Fact	Total Concellation Amt	
Loan Fact	Total Disbursement Amt	
Loan Forgiveness	Amount Diff	Forgiveness Date, Forgiveness Type, Lender Branch, Student, Loan, School Branch, Guaranty Agency, Lender Branch Servicer
Loan Forgiveness	Cumulative Amt	
Loan Refund	Amount Diff	Refund Date, Lender Branch, Student, Loan, School Branch, Guaranty Agency, Lender Branch Servicer
Loan Refund	Cumulative Amt	
Loan Repayment Plan	Term of Repayment in Months	Repayment Plan Type, Repayment Plan Date, Lender Branch, Student, Loan, School Branch, Guaranty Agency, Lender Branch Servicer
Loan Repurchased	Repurchased Amt	Rehabilitation Ind, Repurchased Date, Lender Branch, Student, Loan, School Branch, Guaranty Agency, Lender Branch Servicer
Loan Supplement	Total Reinsurance Claim Payment	Collection Date, Last Int Coll Date, Last Ins Cl Reason, Lender Branch, Student, Loan, School Branch, Guaranty Agency, Lender Branch Servicer
Loan Supplement	Current Supplemental Fee Balance	
Loan Unpaid Refund Discharged	Amount Diff	Adjusted Date, Lender Branch, Student, Loan, School Branch, Guaranty Agency, Lender Branch Servicer
Loan Unpaid Refund Discharged	Cumulative Amt	
Pell Grant	Accepted Grant Amt	Family Contribution Index, Pell Award Year, Pell Grant, Attending School, Reporting School, School Branch, Student
Pell Grant	Cumulative Paid-to-Date	
Pell Grant	Education Cost	
Pell Grant	Remaining Amt to be Paid	
Pell Grant	Scheduled Grant Amt	

Table Name	Fact Name	Entry Level
Reinsurance Claim Payment	Amount Diff	Claim Paid Date, Claim File Date, Reinsurance Rate Cd, Claim Payment Reason Cd, Guaranty Agency, Loan
Reinsurance Claim Payment	Cumulative Req Amt	
Reinsurance Claim Refund	Cumulative Remitted Amt	Refund Date, Guaranty Agency, Loan
SPA Payment	Cumulative Amt	GA Payment Date, Guaranty Agency, Loan
GA Submittal Information	Error Count	Scheduled Submittal Dt, Actual Submittal Dt, Job Completion Dt
GA Submittal Information	Lvl One Error Total	
GA Submittal Information	Lvl Two Error Total	
GA Submittal Information	Lvl Three Error Total	
GA Submittal Information	Error Rate	
GA Submittal Information	Days Late	
GA Submittal Information	Records Processed	
GA Submittal Information	Total Records Lvl1 Proc	
FS Submittal Information	Error Count	Scheduled Submittal Dt, Actual Submittal Dt, Job Completion Dt
FS Submittal Information	Lvl One Error Total	
FS Submittal Information	Lvl Two Error Total	
FS Submittal Information	Lvl Three Error Total	
FS Submittal Information	Error Rate	
FS Submittal Information	Days Late	
FS Submittal Information	Records Processed	
FS Submittal Information	Total Records Lvl1 Proc	
SCH Submittal Information	Error Count	Scheduled Submittal Dt, Actual Submittal Dt, Job Completion Dt
SCH Submittal Information	Lvl One Error Total	
SCH Submittal Information	Lvl Two Error Total	
SCH Submittal Information	Lvl Three Error Total	
SCH Submittal Information	Error Rate	
SCH Submittal Information	Days Late	
SCH Submittal Information	Records Processed	
SCH Submittal Information	Total Records Lvl1 Proc	
Supplemental Reinsurance Payment	Amount Diff	GA Claim Date, GA Claim Reason Cd, Guaranty Agency, Loan
Supplemental Reinsurance Payment	Cumulative Req Amt	
Supplemental Reinsurance Payment	Other Fees	

7.6 Dimensional Model Functionality and Advantages

The dimensional data model has several advantages over the current NSLDS data model. First and foremost, the new data model was designed with a heavy focus on reporting and querying efficiency. The new structure will reduce the join paths required for queries. This efficiency was achieved by defining facts at multiple levels rather than requiring a join to the loan table. While this structure will also support the ETL process, it will require additional transformations to achieve the planned extension of the incoming facts. In addition, this structure will also provide a more efficient access path for data through the NSLDS II web interface.

By examining a sample query the simplification of data relationships between tables can be identified. Consider the following example:

Question: *Which students have had a loan refund? How much was that refund*?

Current Database SQL:

```
SELECT     stu.no, stu.curr_lst, sum(loan_rfd.cum_len_recd_amt)
FROM       loan, loan_rfd, stu
WHERE      loan.no = loan_rfd.loan_no
AND        stu.no = loan.stu_no
AND        stu.seq_no = loan.stu_seq_no
GROUP BY stu.no, stu. curr_lst
```

New Database SQL:

```
SELECT     student.stu_id, student.curr_lstnm,
sum(loan_refund.cum_len_recd_amt)
FROM       loan_refund, student
WHERE      student.stu_id = loan_refund.stu_id
AND        student.stu_seq_no = loan_refund.stu_seq_no
GROUP BY  student.stu_id, student.curr_lastname
```

Notice in the first query that the loan table is needed for this analysis because the loan refund table does not have a direct relationship with the student even though a single student owns the loan. The second query does not require a join to the loan table. The loan refund table has a direct relationship to the student table via the student ID. This is why the join to the loan table is not necessary.

The following table indicates the number of rows in each of the tables used in the preceding queries:

Table	Row Count (as of 6/2002)
LOAN	144,476,234
LOAN REFUND	15,421,135

STUDENT	46,249,164

Table 5, Row Counts

Because the first query required a join to the loan table, which has over 140 million rows, the query takes up valuable processing time. The more joins and table rows in each table, the more time and resources required of the database to execute a query. With the implementation of the new data model, the second query does not require the join to the loan table, decreasing processing time and overall usage of database resources.

Lastly, there are additional relational entities between dimensions called associative entities. This will allow a user to drill across dimensions without interacting, in a significant manner with large fact tables. For example, lenders will have access to student information without going through the fact or loan tables.

8 NSLDS II Data Access Approach

The data access approach describes the data access tool that will be used to retrieve information from the reengineered NSLDS database. The approach examines the legacy NSLDS data access tools and provides an overview of new data access tool that will be used with the reengineered NSLDS.

The data access tool provides the mechanisms and architecture to access and display data in an understandable and flexible way to the end user. Access methods include query and reporting tools, online analytical processing (OLAP) tools, and data mining and knowledge discovery.

FSA has identified MicroStrategy as the standard data access platform to be used for enterprise data management projects (FSA Technology Policies, Standards & Products Guide). The data access approach discusses how NSLDS II will use MicroStrategy to manage data access to provide high quality business intelligence for its users.

8.1 MicroStrategy Platform Architecture

The MicroStrategy platform architecture is made up of the several products. These products provide end user functionality, power for developers, and control for administrators.

- MicroStrategy Intelligence Server
- MicroStrategy Desktop
- MicroStrategy Web
- MicroStrategy Architect
- MicroStrategy Administrator
- MicroStrategy SDK

MicroStrategy delivers services through the use of Microsoft Windows workstations and standard web browsers. The figure below illustrates the MicroStrategy product platform architecture divided by functionality.

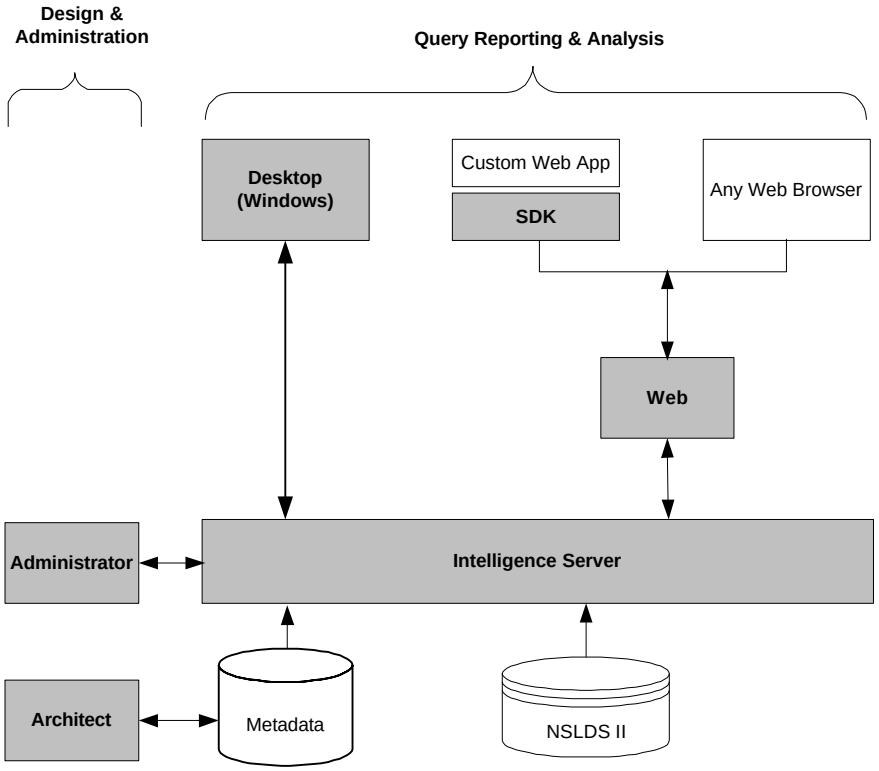


Figure 13, MicroStrategy Platform Architecture

MicroStrategy Intelligence Server

The MicroStrategy Intelligence Server uses the centralized metadata repository to convert report requests into optimized multi-pass SQL queries and performs any additional calculations not available in the database. The Intelligence Server is responsible for controlling all interactions with the NSLDS and provides a central point of security and administrative control.

MicroStrategy Desktop

The MicroStrategy Desktop is the desktop tool used for query and reporting, data analysis, and report development. The desktop application will primarily be used by the application developers. MicroStrategy Desktop submits all requests directly to the MicroStrategy Intelligence Server, which translates the requests into database queries.

MicroStrategy Web

MicroStrategy Web allows for report viewing, formatting, drilling, ad hoc querying, and report generation and design from a Web browser. HTML/DHTML interfaces are used to deliver this functionality without using any ActiveX or Java downloads, which should allow MicroStrategy Web to work through standard firewalls.

MicroStrategy Architect
MicroStrategy Architect is a rapid development tool that maps the physical structure of the database into a logical, object-oriented, model of the business that is stored in a centralized metadata repository. The logical business model abstraction of the physical database makes the report design faster and more intuitive. The object-oriented nature of the metadata repository allows changes in the logical model to propagate immediately and transparently to all reports and users of those objects.

MicroStrategy Administrator
MicroStrategy Administrator is the management toolset for administering and maintaining security and the system configuration as well as deploying reporting applications from the development to production environments. The Administrator tool can be used to for analyzing database usage, report usage, user statistics, and for performance tuning and resource planning.

8.2 Metadata

MicroStrategy has two major layers of metadata objects, each layer representing a higher level of abstraction of the NSLDS II physical database. Schema objects are the highest layer and relate structures in the reengineered NSLDS to understandable business terms. Application objects, the second type of metadata, use schema objects to build reports and metrics.

The MicroStrategy Architecture tool defines schema objects by automatically interpreting the NSLDS physical data model and logical business model and portrays them in two graphical views.

- The physical view displays the table structure, as tables, columns, and data types.
- The logical view displays the attributes and facts represented in each table and relationships between tables, as defined in the multidimensional view stored in the metadata repository.

8.3 User Interfaces

MicroStrategy allows data access either through a Web browser or through a Windows Desktop application. Most users will retrieve data from reports using the Web interface. MicroStrategy also provides wizards on both the web and desktop interfaces to guide users through building reports, developing ad hoc queries, and customizing drill paths.

Using drag-and-drop mechanics users can manipulate the data on a report and save the data as a new report. MicroStrategy supports the following data manipulation:

- Sorting – Sort on any metric directly on a report
- Pivoting – Re-arrange the order and placement of attributes and metrics on reports
- Page-by – Allows a subset of the report to be created grouped by specific data elements
- Drill down – Increase the level of detail of a summary level report
- Drill anywhere – Allows users to drill down any path regardless of whether a metric is on a report

- Total/subtotal – Add totals or subtotals to metrics on a report
- Derived metrics – Create new metrics as new columns on a report based on existing metrics in the report

MicroStrategy also provides users the ability to format reports by changing the font, number format, as well as other attributes. Once a report has been generated, the user can save, print, or export the report. These report actions can be performed from either the Web or Windows interface.

8.4 MicroStrategy Intelligence Server

The MicroStrategy Intelligence Server is the architectural foundation of the platform and it performs the analytical, performance, administrative, and security functions. The Intelligence Server dynamically process queries from the user interfaces. The main components of the MicroStrategy Intelligence Server are:

- Intelligence server bus
- Communications Layer for metadata and the data warehouse
- SQL generation engine
- Query Engine
- Analytical engine

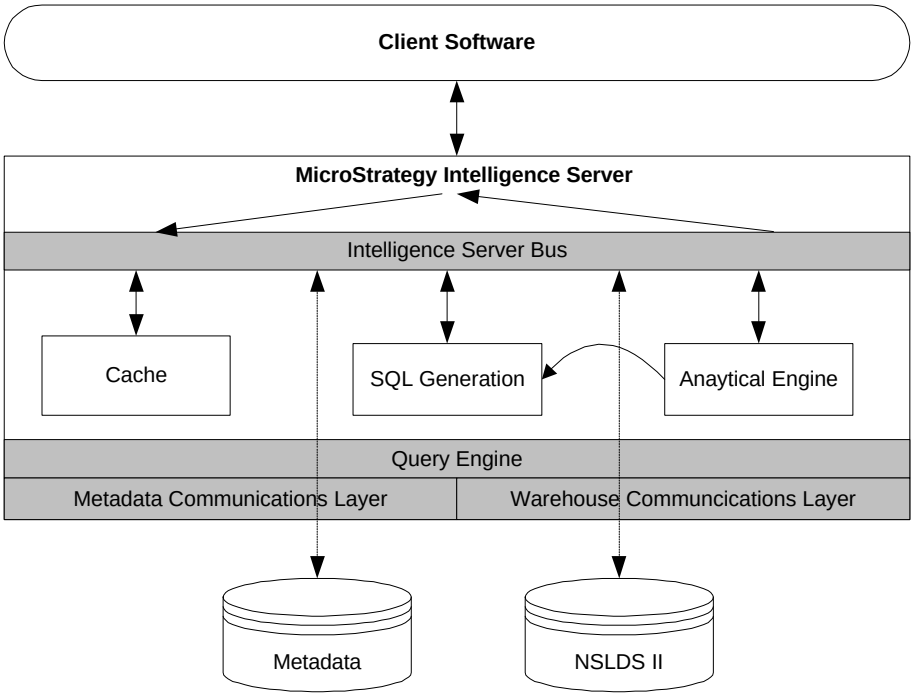


Figure 14, MicroStrategy Intelligence Server

The process steps listed below demonstrate how the MicroStrategy Intelligence Server handles report requests.

- The MicroStrategy Intelligence Server receives a report request from the client software, which is passed to the Intelligence Server bus.
- The Intelligence server checks to see whether the report has already been run. If the report has not be cached, the Intelligence Server bus obtains the report definition and application objects from the metadata.
    - If the report has already been cached, the results are returned to Intelligence Server bus and the bus passes the formatted results back to the client software.
- The bus sends this information to the SQL Generation Engine, which creates an optimized SQL query and returns the SQL to the bus.
- The bus sends the SQL to the Query Engine, which runs the SQL against the data warehouse and reports the results to the bus.
- The Intelligence Server bus invokes the Analytic Engine and in the case of a complex report may pass the results back to the SQL Generation Engine for more processing. The final results of the query are returned to Intelligence Server bus.
- The Intelligence Server bus passes the formatted results back to the client software.

By processing all report requests through the Intelligence Server, MicroStrategy is able to centralize communication and maximize security. The Intelligence Server is the only component of this architecture that requires an Open Database Connectivity (ODBC) connection to NSLDS II.

Additionally, the Intelligence Server is able to generate SQL code that is optimized for the NSLDS II database, which is equivalent if not superior to the best-handcrafted SQL. Users create ad hoc queries using the drag-and-drop interface and the MicroStrategy SQL Generator creates the code. MicroStrategy can also leverage over 90 Very Large Database (VLDB) driver settings that alter the syntax and processing of the SQL specifically for the NSLDS II database management software.

MicroStrategy Intelligence Server also uses multi-level caching to increase performance. When the Intelligence Server retrieves the requested data from NSLDS, it creates Intelligent Cubes that allow users to perform additional analyses on cached data without having to re-access the database. Almost any form of data manipulation can be performed on the cached report without the need to query the database again. Caches are automatically updated whenever an underlying object of a particular report is updated.

8.5 MicroStrategy Security

MicroStrategy integrates with a wide variety of security products and methods. MicroStrategy security is divided into the following five main areas that provide a comprehensive security at every layer of the platform.

- User authentication

- Application functionality security
- Access control lists
- Data security
- Internet architecture and transmission security

MicroStrategy maintains a profile for each authorized user of the platform and each user is prompted to enter a password, which is used to authenticate the user. Users fall into various types ranging from causal to power users. Depending on the user's needs, certain privileges will be assigned to user or user group. Security roles are used to group together a specific combination of privileges, which are then assigned to each user. These privileges limit the specific functions and action each user can perform.

Access control lists (ACL) consisting of users and/or user groups as well as permissions granted to those users control access to very object. Permissions can be assigned to reports, templates, filters, and metrics. On a report containing five metrics, each metric's ACL determines whether a user can view the metrics in the report. Permissions can be granted to include:

- Browse
- User
- Execute
- Read
- Write
- Delete
- Control

MicroStrategy allows for restrictions to be placed on tables, rows, and columns with those tables by using security filters tied to the user group and permissions. This results in cell level data security.

MicroStrategy also supports Internet architecture and transmission security by incorporating 3rd generation multi-tier architecture as its foundation.

- The MicroStrategy Intelligence Server, which manages database connections, resides behind two firewalls.
- MicroStrategy Web allows administrator to configure which port is used for data access allowing intrusive port scans to blocked by the firewall.
- The use of External Remote Procedure Calls (RPC) or Remote Method Invocation (RMI) is not permitted.
- MicroStrategy does not use any downloadable applets that can be intercepted by the end user's firewall.

Lastly, MicroStrategy takes advantage of data encryption and transmission. Data is encrypted as it flows through the system and encryption can be used to encode data as it is being

transmitted across the network or while it is being stored. MicroStrategy uses following encryption protocols:

- Secure Socket layer (SSL)
- Tiny Encryption Algorithm (TEA)
- RACE Integrity Primitives Evaluation MD-160 (RIPEMD-160)

8.6 Assumptions

- GA and school users will access NSLDS II using the MicroStrategy Web interface.
- Instead of direct SQL, MicroStrategy objects will be used to access NSLDS II.

8.7 Risks and Contingencies

Risk	Mitigation Strategy
GA and schools will not be able to access NSLDS II using the MicroStrategy Web interface.	GA and schools will receive NSLDS II reports through SAIG.
Users will no longer be able to access NSLDS II using direct SQL code.	Train users to create queries and reports using MicroStrategy objects.

9 NSLDS II Development Architecture Approach

The Development Architecture is a unified collection of technology services, tools, procedures and standards for constructing and maintaining application software.

The purpose of the development environment is to support the tasks involved in the analysis, design, construction and maintenance of business systems, as well as the associated management processes. It is important to note that the environment should adequately support *all* the development tasks, not just the code / compile / test / debug cycle. Given this, a comprehensive framework for understanding the requirements of the development environment should be used.

The development architecture is designed to mimic the production environment in every possible way, given cost constraints. Since the development environment is not required to handle the volume of traffic of the production systems, and since the database does not need to be sized for production in the development environment, much of the architecture has been scaled back with respect to number of processors and amount of memory and storage. However, wherever possible, the number of servers and intersystem connections has been planned in a production-like manner.

9.1 Technical Assumptions

The following Assumptions are made with respect to the Development Architecture:

1. The Production NSLDS II System will house approximately 4-10 terabytes (TB) of data.

2. The NSLDS II System will use IBM DB2 EEE as its database product, housed on an RS/6000 system. With this decision in mind, the NSLDS II System will be hosted at the Department of Education's Virtual Data Center (hereafter VDC), the data acquisition component will run on HP-UX Unix Servers and the data access component will run on Windows NT Servers.
3. Connectivity between the Washington DC Modernization Partner Offices and the VDC has been established. Either this connectivity will need to be extended to the 7th floor of the 820 First Street NE Building, or the NSLDS II Development team will need to be relocated to the Lower Level of the 830 First Street NE Building.
4. The Consistent Answers system will query NSLDS I via the EAI Bus with real-time MQSeries traffic prior to the NSLDS II Implementation. Therefore, in the replatforming effort, the NSLDS II team will focus only on the NSLDS II side of the interfaces.
5. The NSLDS II project has a goal of eliminating the use of magnetic tape in its interface design. Should diplomatic attempts to remove tape from the architecture fail, the NSLDS II architecture needs to account for an interim tape access solution.
6. Rational ClearCase will be the Configuration Management solution.
7. The NSLDS II development team will require two workstations at each developer's desk. The first workstation will access the VDC via a static IP Addresses (this workstation can be the developer's laptop). The second workstation will access the Rational ClearCase domain in Washington DC.

Any changes to the above listed assumptions will require the Technical Architecture team to revisit the Development, Execution and Operations Architecture.

9.2 Infrastructure Requirements

In order to effectively develop the technical portion of the NSLDS II system, developers will require an infrastructure that allows them to configure the system in the most efficient manner while, at the same time, maintaining a parallelism in the architecture to the planned Execution and Operations Architectures. To meet this goal, the NSLDS II development environment will be comprised of Runtime, and Development regions for each of the four functional components of the system: Data Acquisition, Data Storage, Data Access and Web Presentation. Development regions are where the developers themselves will perform their functions in configuring, coding and otherwise implementing the NSLDS II Reengineered system. Runtime regions are where the developer's code is moved to be unit and assembly tested. Due to the selection of Informatica as the Extract Transform and Load (ETL) tool, the data acquisition segment of the system will require an additional layer since Informatica requires a broker where transformation rules are staged before being promoted to a runtime state.

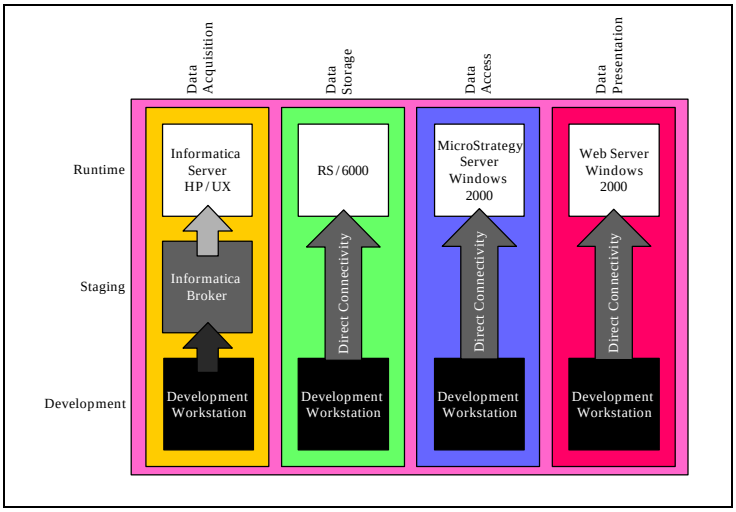


Figure 15, Functional Layout of NSLDS II Development Architecture

In order to create this development environment, there are a number of technical requirements that need to be met. The technical side of this architecture requires planning in the areas of system access, system connectivity, software installation and software configuration.

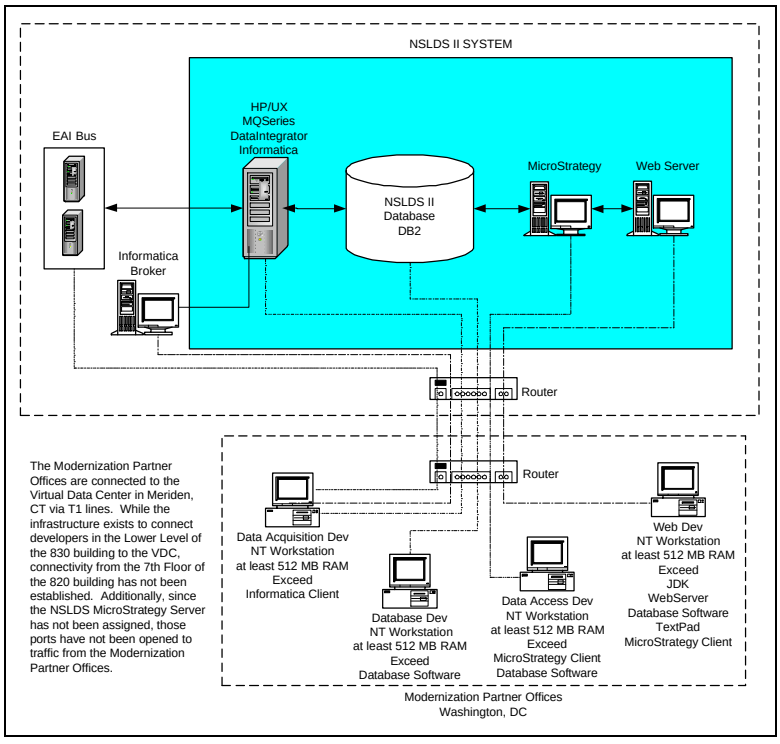


Figure 16, Technical Layout of NSLDS II Development Architecture

9.2.1 User ID's

The FSA Personnel Security Process will need to be followed, and login ID's established for every system with which a user will interface.

- Data Acquisition Developers will need access to each of the interface partners, as well as the Development/Test EAI Bus servers, the NSLDS II Runtime Informatica Server and the NSLDS II Informatica Broker Server.
- Data Storage Developers will need access to the NSLDS II Runtime Informatica Server, the NSLDS II Informatica Broker Server, and the clustered IBM pSeries 660 machine.
- Data Access Developers will need access to the clustered IBM pSeries 660 machine, the MicroStrategy Server and the Web Server.
- Web Interface Developers will need access to the clustered IBM pSeries 660 machine, the MicroStrategy Server and the Web Server.

9.2.2 Intersystem Connectivity

As indicated by figure 5.2, connectivity between a number of systems needs to be established in order to complete the creation of the NSLDS II Development Architecture. The following matrix indicates the connectivity requirements of systems within the development environment.

	EAI Bus Runtime Servers	Data Access Development Workstation	Informatica Broker	Informatica Runtime Server	Data Storage Development Workstation	RS/6000 Database Server	Data Access Development Workstation	MicroStrategy Runtime Server	Presentation Layer Development Workstation	Presentation Layer Runtime Server
EAI Bus Runtime Servers		X		X						
Data Access Development Workstation	X		X	X						
Informatica Broker		X		X		X				
Informatica Runtime Server	X	X	X		X	X				
Enterprise Management Development Workstation				X		X				
Database Server			X	X	X			X	X	
Data Access Development Workstation								X		
MicroStrategy Runtime Server						X	X		X	X
Presentation Layer Development Workstation						X		X		X
Presentation Layer Runtime Server								X	X	

Table 6, NSLDS II Development Architecture Intersystem Connectivity Requirements

9.2.3 Developer Workstations

Below are the requirements for developers on the NSLDS II Project:

- Data Acquisition Developers will require systems with the following minimum resources:

System Function	Data Acquisition Development Workstation
Hardware	TBD
Memory	512 MB
Operating System	Windows NT/Windows 2000
Software	Informatica Client IBM AMI Configuration Client V1.2 IBM MQSeries Exceed or similar Telnet/FTP Package Java JDK Textpad
Location	Mod Partner -- Washington, D.C.

- Data Storage Developers will require systems with the following minimum resources:

System Function	Data Storage Development Workstation
Hardware	TBD
Memory	512 MB
Operating System	Windows NT/Windows 2000
Software	Database Design Tool Database Configuration Tool Exceed or similar Telnet/FTP Package Textpad
Location	Mod Partner -- Washington, D.C.

- Data Access Developers will require systems with the following minimum resources:

System Function	Data Access Development Workstation
Hardware	TBD
Memory	512 MB
Operating System	Windows NT/Windows 2000
Software	MicroStrategy Client Software?? Exceed or similar Telnet/FTP Package Database Software (for coding/unit testing) Textpad
Location	Mod Partner -- Washington, D.C.

- Data Presentation Developers will require systems with the following minimum resources:

System Function	Data Presentation Development Workstation
Hardware	TBD
Memory	512 MB
Operating System	Windows NT / Windows 2000
Software	Exceed or similar Telnet / FTP Package JDK Web Server Software (for coding / unit testing) Database Software (for coding / unit testing) Textpad Web Design Software (Frontpage / Dreamweaver)
Location	Mod Partner -- Washington, D.C.

9.2.4 EAI Bus Dev / Test Servers

Connectivity to be established between NSLDS II Development Data Acquisition Server and the EAI Dev / Test Servers
Channels, Queues, etc. pointing to and from NSLDS II Development Server
The configuration details of the EAI Bus Dev / Test Servers is as follows:

System Function	EAI Bus
Hardware	2 Sun 3500 Enterprise Servers
Memory	4 GB / server
Processors	4X400Mhz Sun UltraSparc / server
Operating System	Solaris 7 (64-bit)
Software	IBM MQSeries 5.1 IBM AMI 1.2 CommerceQuest DataIntegrator 4.2.4
Location	VDC -- Meriden, CT

9.2.5 NSLDS II Development Data Acquisition Runtime Server

Informatica, the ETL Tool selected for the NSLDS II solution, runs best in a Unix environment. Due to this fact, the NSLDS II Development Data Acquisition Runtime Server will be an HP-UX system built with the following considerations:

System Function	Informatica Runtime Server
Hardware	HP L-Class
Memory	4 GB
Processors	4 X 440 Mhz
Operating System	HP-UX
Software	IBM MQSeries 5.2 IBM AMI 1.2 CommerceQuest DataIntegrator 4.2.4 Informatica PowerConnect 5.0 JDK and JRE 1.2 or greater (for custom Adapters)
Location	VDC -- Meriden, CT

9.2.6 NSLDS II Development Data Acquisition Staging Server

In order to support development runtime architecture similar to the eventual operations architecture, a Data Acquisition Staging Server is required with the following specifications:

System Function	Informatica Broker
Hardware	Compaq DL380
Memory	2 GB
Processors	2 X 866 Mhz Pentium III
Operating System	Windows NT/Windows 2000
Software	Informatica PowerConnect Broker
Location	VDC -- Meriden, CT

9.2.7 NSLDS II Development Database Server

The NSLDS II Development Database Server will run on a 4-way pSeries 660 Model 6H1 server with 4GB of memory and four fibre channel adapters (two for disk and two for tape). It will be connected to its own 2.4TB FAStT Storage Array via two fibre channel connections.

This Development environment will also connect into a fibre channel switch fabric. This switch fabric enable will enable the server to also utilize the 3584 LTO Tape Library for tape backup using Tivoli Storage Manager (TSM).

The pSeries 660 Model 6H1 server will run with the following configuration:

System Function	Data Storage
Hardware	pSeries 660 Model 6H1
Memory	4 GB
Processors	4 x 750 MHz
Operating System	IBM AIX
Software	DB2 EEE
Location	VDC -- Meriden, CT

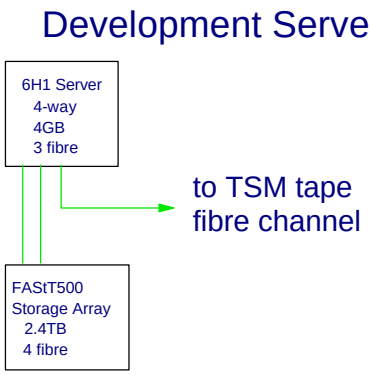


Figure 17, Data Storage Development Server Configuration

9.2.8 NSLDS II Development Data Access Application Server

MicroStrategy, the reporting/data access tool selected for the NSLDS II solution, runs best in a Windows environment. Due to this fact, the NSLDS II Development Data Access Server will be a Windows system built with the following considerations:

System Function	MicroStrategy Server
Hardware	Compaq DL380
Memory	2 GB
Processors	2 X 866 Mhz Pentium III
Operating System	Windows NT/Windows 2000
Software	MicroStrategy -- PRODUCT???
Location	VDC -- Meriden, CT

9.2.9 NSLDS II Development Data Access Web Server

Need Introduction. Need to confirm platform of Web Server.

System Function	Web Server (Staging and RunTime)
Hardware	Compaq DL380
Memory	2 GB
Processors	2 X 866 Mhz Pentium III
Operating System	Windows NT/Windows 2000
Software	IIS
Location	VDC -- Meriden, CT

10 NSLDS II Execution Architecture Approach

The execution environment is a unified collection of run-time technology services, control structures, and supporting infrastructure upon which application software runs.
The execution architecture is divided into two sections. The first section is the architecture and processes by which the initial data load of NSLDS II will be conducted through a snapshot of the legacy NSLDS system. Following the Data Migration Architecture is the day-to-day Production Architecture, a section that details the batch architecture and access architecture.

10.1 Technical Assumptions

In addition to the technical assumptions outlined under the Development Architecture heading, the following technical assumptions are made with respect to the Execution Architecture:

- To process files transferred to NSLDS II from interface partners, an Informatica script will need to be triggered by the DataIntegrator product.
- To transfer files from NSLDS II to interface partners, a DataIntegrator script will need to be triggered by the Informatica product.
- The Informatica native scheduler incorporates sufficient functionality to meet all file creation requirements of the NSLDS II.
- The MicroStrategy native scheduler incorporates sufficient functionality to meet all report creation requirements of the NSLDS II.

10.2 Production Data Storage Architecture

The initial production hardware infrastructure (illustrated below), designed to support 4TB of raw data, is comprised of integrated server and storage components. These include:

- An eServer Cluster 1600 system, with four cluster-ready pSeries 660 Model 6M1 servers connected with a high speed interconnect (the SP Switch2) and managed by a pSeries 610 Model 6E1 Control Workstation
- Four 3TB FAStT500 storage arrays

The server environment will run the AIX (UNIX) operating system and IBM's clustering software, Parallel Systems Support Program (PSSP), which enables the cluster to be managed as a single system. For completeness, the solution also includes a backup environment comprised of a pSeries 660 Model 6H1 server running Tivoli Storage Manager (TSM) for automated tape backup and an LTO Tape Library system with four LTO tape drives. This hardware configuration will be covered in more detail in the Operations Architecture delivered as part of the Detailed Design phase of NSLDS II Reengineering.

The production DB2 EEE data warehouse will run on the interconnected 6M1 servers. Each of these servers is configured with eight 750 MHz copper based processors and 16GB of memory. Each 6M1 server features six fibre channel adapters (four for disk and two for tape) distributed across two I/O drawers. This multiple I/O drawer configuration ensures sufficient server I/O bandwidth to accommodate the six fibre channel adapters and provide a balanced system design with I/O in step with processing capacity.

All production server components will be interconnected using the SP Switch2 high speed interconnect technology. This highly available interconnect, featuring redundant components and current maintenance, as well as multiple paths with automatic re-routing, provides a bi-directional bandwidth of 1 GB/sec between servers. More importantly, this bandwidth will scale proportionally as additional servers are added to the cluster.

The storage component of the data warehouse architecture includes four FAStT500 Storage Arrays configured with four fibre channel connections and a useable capacity of 3TB each, for an overall useable capacity of 12TB for the production warehouse environment. The amount of storage in each array was limited, by design, to 3TB to maximize the I/O throughput of the storage configuration. The FAStT employs a RAID-5 implementation. The total proposed useable disk storage capacity of 12TB provides for three times the amount of raw data. This will provide ample room for indices, temp space, any desired aggregation, as well as staging area.

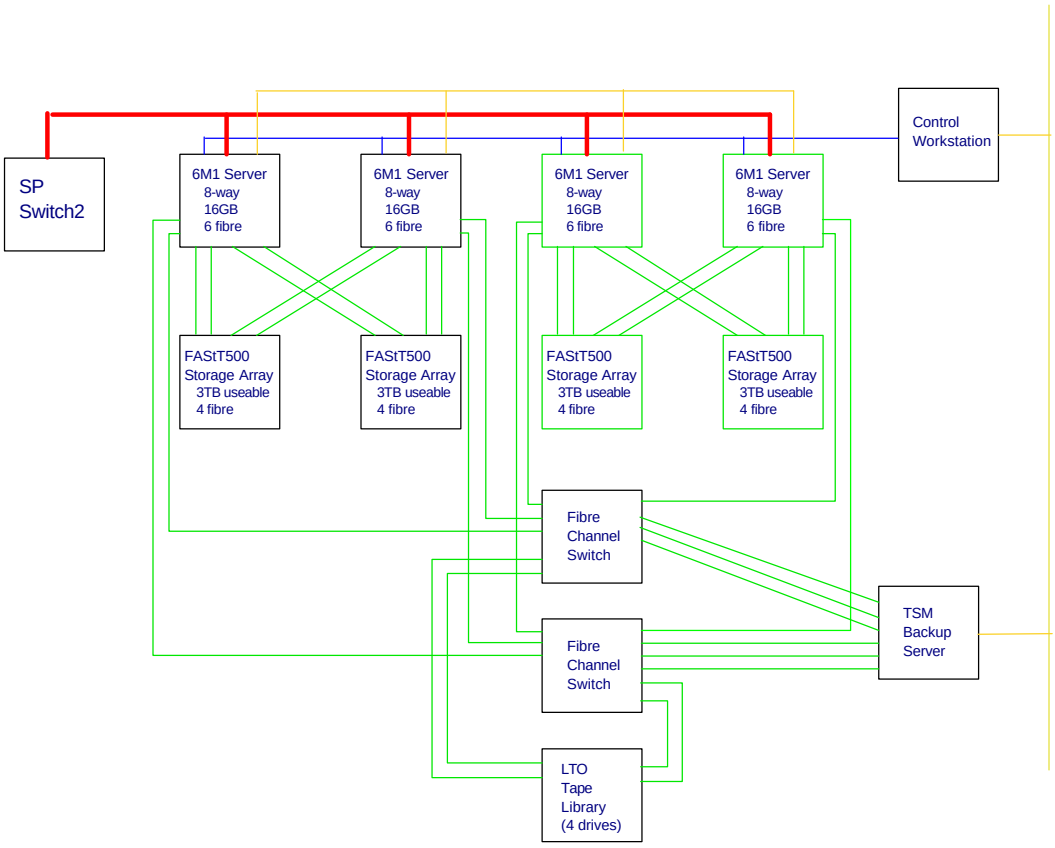


Figure 18, Data Storage Production Server Configuration

10.3 Data Migration Execution Architecture

This section provides the technical plan for migrating data from the legacy NSLDS to the reengineered NSLDS II. This section is the technical complement to the document entitled, Data Conversion and Migration Strategy.

The process of migrating the legacy database to the target NSLDS II server is, by necessity, an extremely manual process. To that end, the use of scheduling tools is unnecessary, as each step of the process must be validated and confirmed before continuing to the next step.

10.3.1 Legacy NSLDS Architecture Changes

As described in the Data Acquisition Approach section of this document, Data Migration will require the establishment of a parallel set of core database tables for the Legacy NSLDS. These tables will exist in a mainframe environment that can be accessed by the instance of Informatica PowerConnect for Mainframes that is highlighted in the Data Acquisition Approach. Informatica connectivity will need to be established between this environment and the NSLDS II Informatica Server for successful data migration to take place.

10.3.2 NSLDS II Architecture

The NSLDS II Informatica Server will need to have connectivity established with three systems. First, the NSLDS II Informatica server will require Informatica connectivity to the legacy NSLDS environment hosting the database instance, as mentioned in the previous section. Next, the NSLDS II Informatica Server will require connectivity to the NSLDS II database, to write the results of completed data migration and conversion processes. Lastly, the NSLDS II Informatica Server will require connectivity to the repository that is housing NSLDS-bound traffic stored after the legacy NSLDS makes it's snapshot for data migration purposes. This traffic will need to be processed after the data migration process is completed, but prior to accepting live interface traffic.

10.4 Production Execution Architecture

Once the Data Migration process has been completed, and all suspended interfaces have been processed, production execution will begin. The execution architecture for production includes processes for FSA system, school, Guaranty Agency and clearinghouse batch interfaces, as well as end-user reports and other data access points.

10.4.1 Batch Architecture

Batch interfaces with the reengineered NSLDS II can be divided into two categories with respect to batch processing and scheduling: Externally Initiated and Internally Initiated. An example of an externally initiated interface is one in which an interfacing system submits a file to NSLDS II. For such an interface, the interfacing system may or may not expect a response or error file. An internally initiated interface is one in which NSLDS II creates the first batch file and sends it to another entity. In such a case, the transmission also may or may not have a response or error file.

The execution architecture for externally initiated interfaces will take advantage of the scheduling capabilities of the DataIntegrator Tool. Once DataIntegrator finishes delivering a file to NSLDS II to be processed, it can, upon successful delivery, perform operations on the target system including, but not limited to, calling scripts. Therefore, part of the DataIntegrator process of delivering files to NSLDS II will be the task of calling the Informatica job at NSLDS II to handle the file according to defined business rules. Processing a file as soon as it arrives, rather than waiting for a specific, time-based window in which to process enhances NSLDS II's ability to provide the most current data possible to downstream interface partners.

The Informatica job that processes incoming files will end, where necessary, by triggering another Informatica job that produces a response or error file. Once the creation of the response / error file is complete, the Informatica job will call a DataIntegrator script that will transfer the response file to the interface partner.

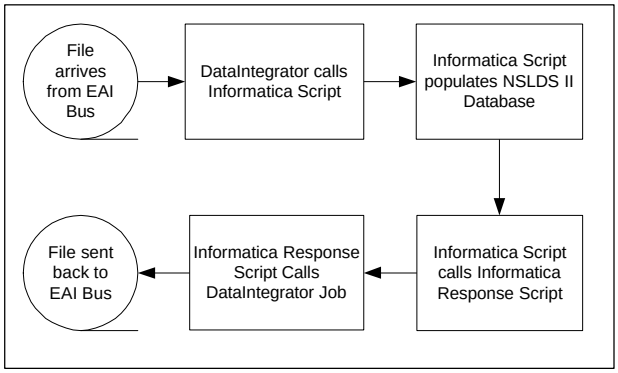


Figure19, Externally Initiated Interface Scheduling

For internally initiated interfaces, jobs will originate in the Informatica scheduler. When the scheduling criteria is met, Informatica will begin to produce a file to be sent to an interface partner. Once the file has been created, Informatica will call a DataIntegrator script to send the file to the interface partner in a similar fashion to the response to the externally initiated interface.

In situations where there is a response to an internally initiated interface, there is the possibility of a significant lag time between the creation / sending of the interface file and the receipt of a response or error file. Since NSLDS II should not hold up other processing while waiting for this response, these responses will be processed asynchronously from the original file. Therefore, sending the output file to the trading partner will happen at the end of the job.

If a response or error file is received at NSLDS II, such files will be handled in exactly the same manner as externally initiated files that do not require responses. That is, DataIntegrator will trigger an Informatica Script upon successful completion of the delivery process. Informatica will process data in the response / error file.

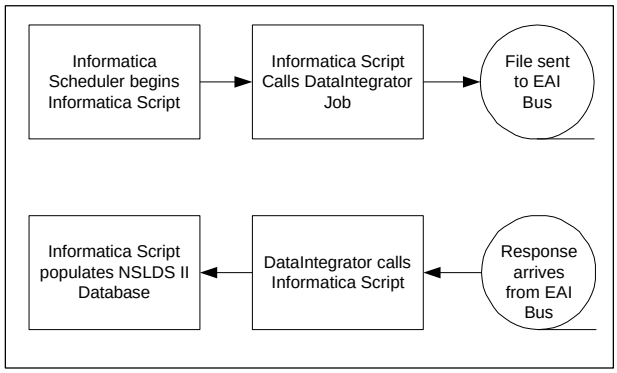


Figure20, Internally Initiated Interface Scheduling

10.4.2 Access Architecture

Data access consists of two types of access: real-time queries, and scheduled reports. Due to the *ad hoc* nature of real-time querying, there is no need to schedule such interfaces. The scheduling of reports will be done in the scheduler included with the MicroStrategy product.

Appendix A: Logical Dimensional Model

Logical Dimensions

A line between tables represents a relationship of the type one-to-one (1:1, no arrows at either end), one-to-many (1:M, a single arrow at one end), or many-to-many (M:M, arrows at both ends). Groupings of tables encapsulated by a brace ({) show that each table in the group has the same relationship as directed by the lines.

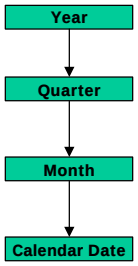
Key

1:M

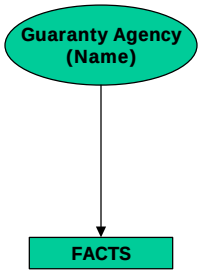
M:M

1:1

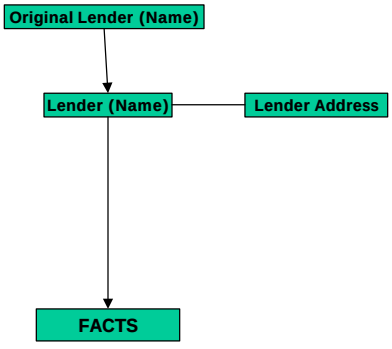
Generic Calendar



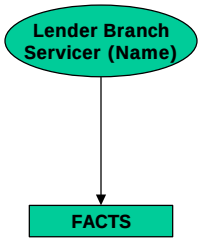
Guaranty Agency



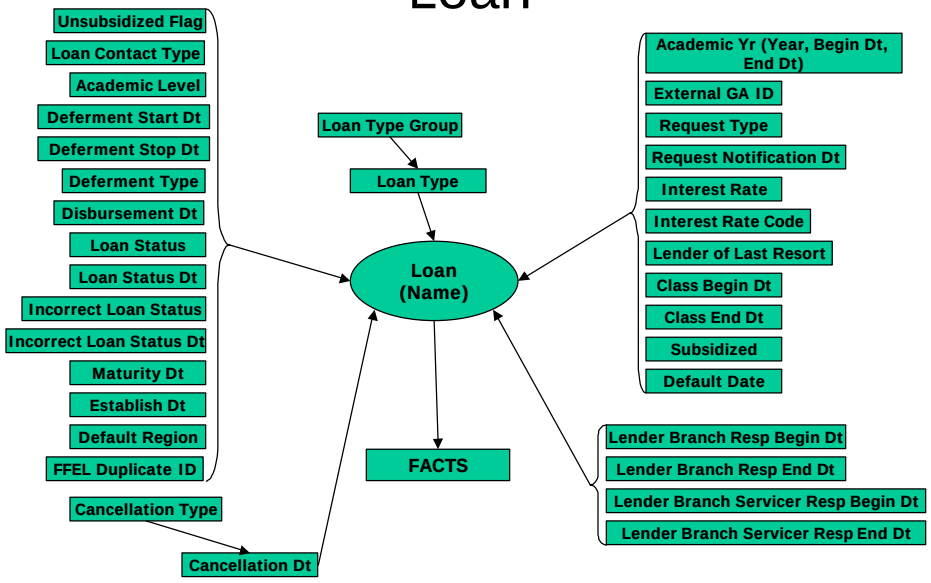
Lender



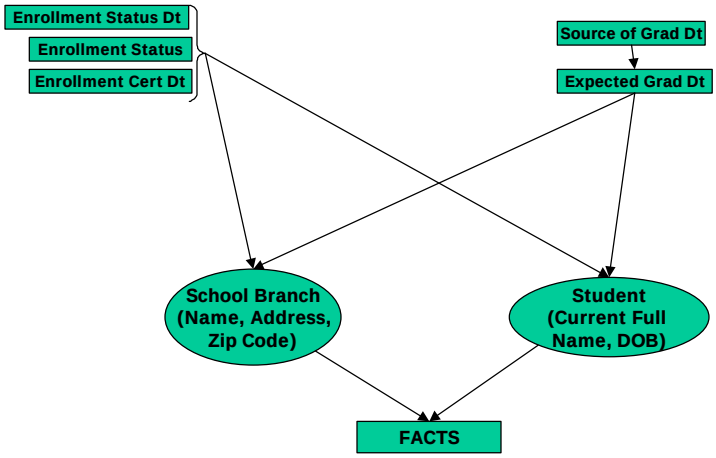
Lender Branch Servicer



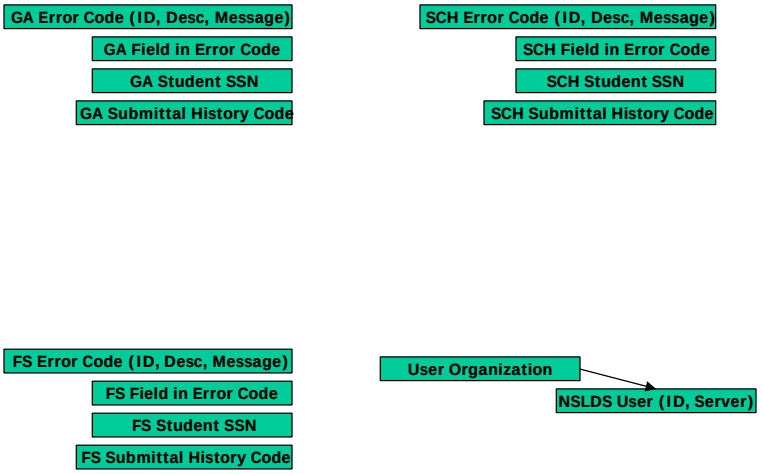
Loan



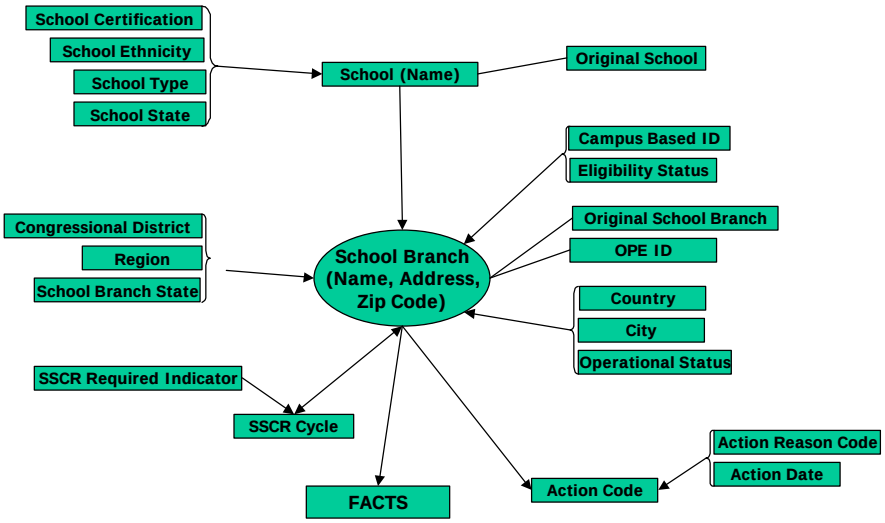
Miscellaneous



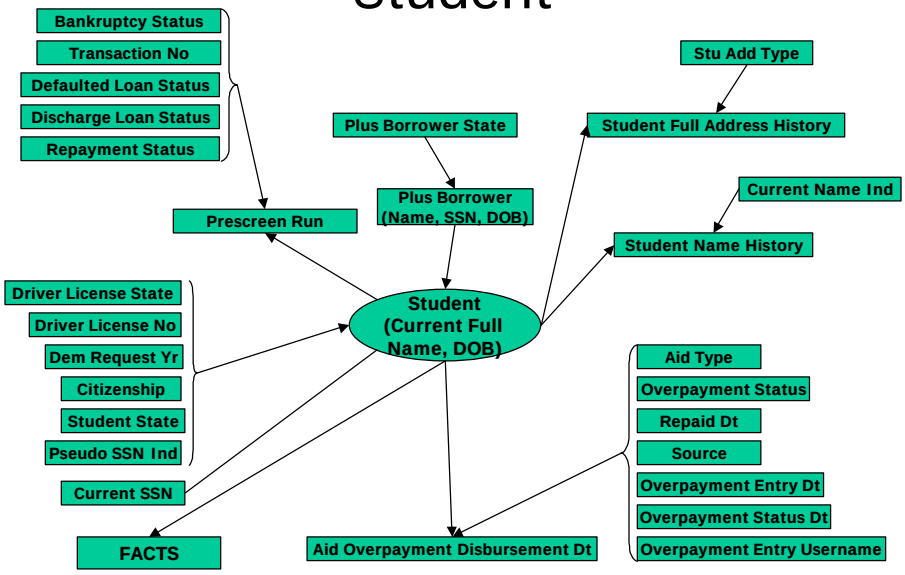
Miscellaneous



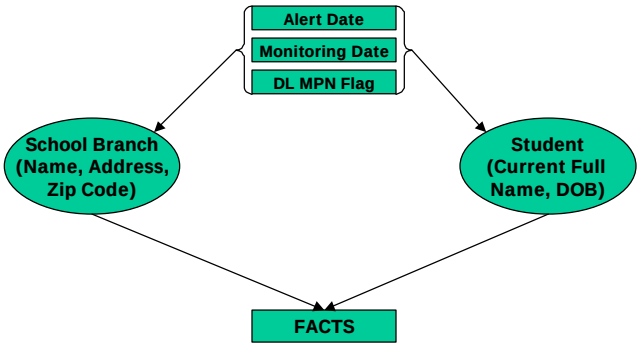
School



Student

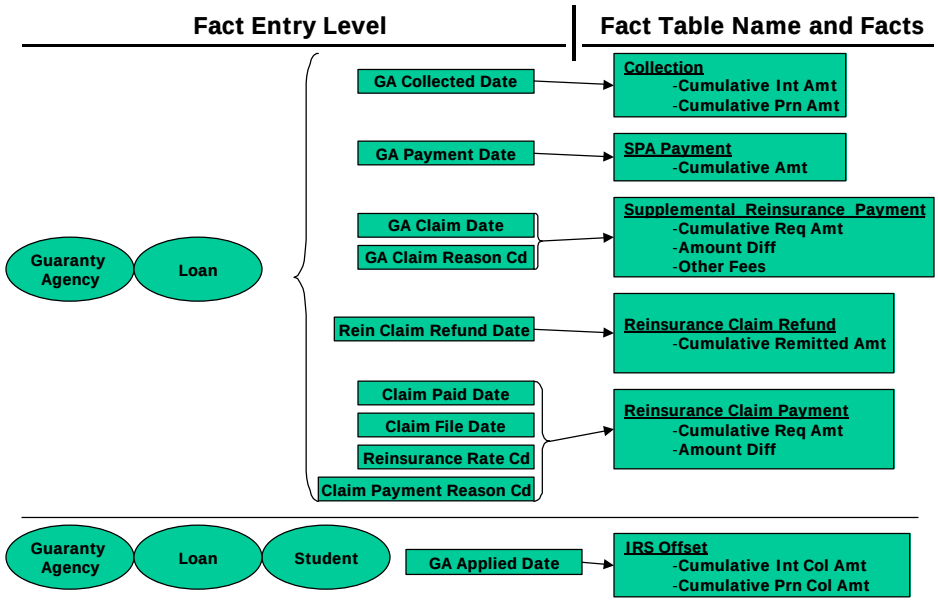


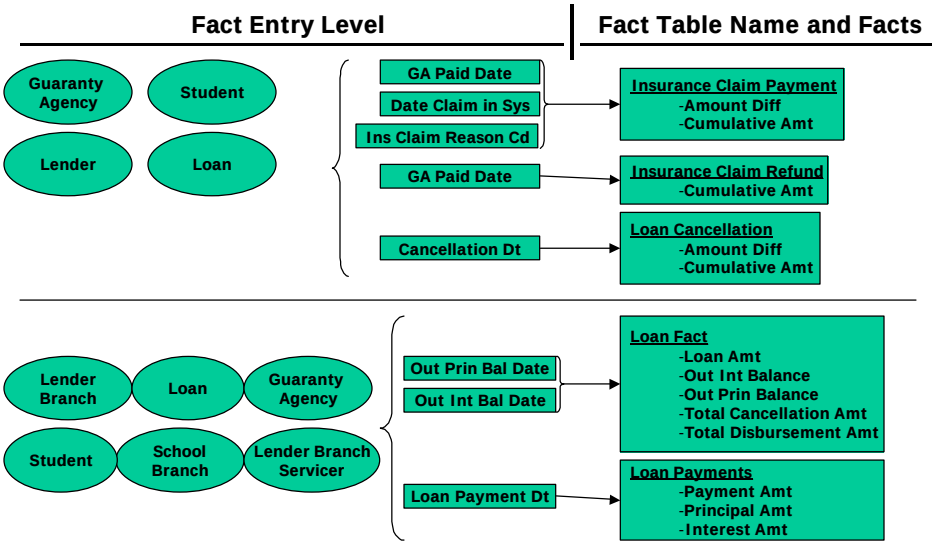
Transfer Monitor

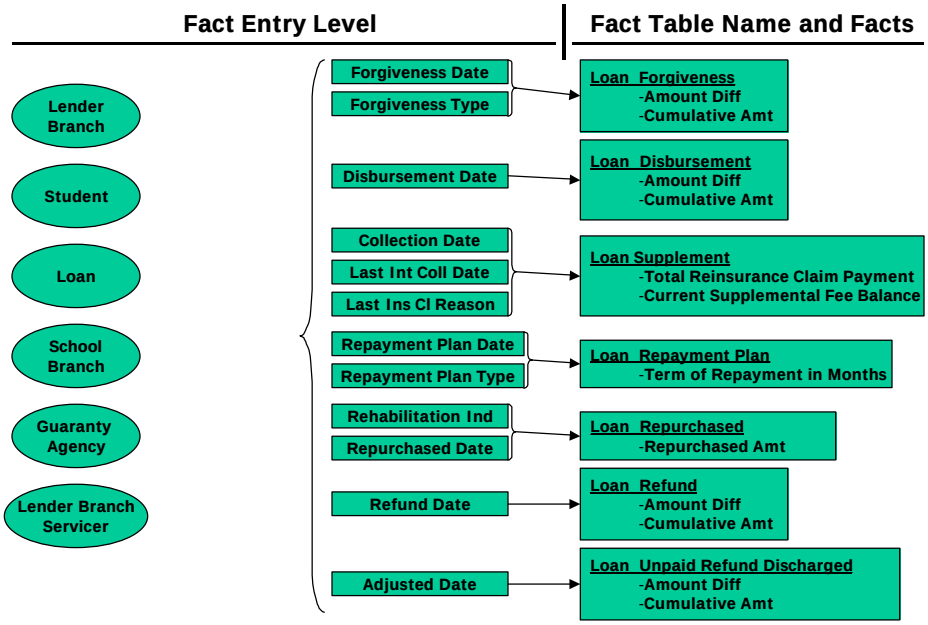


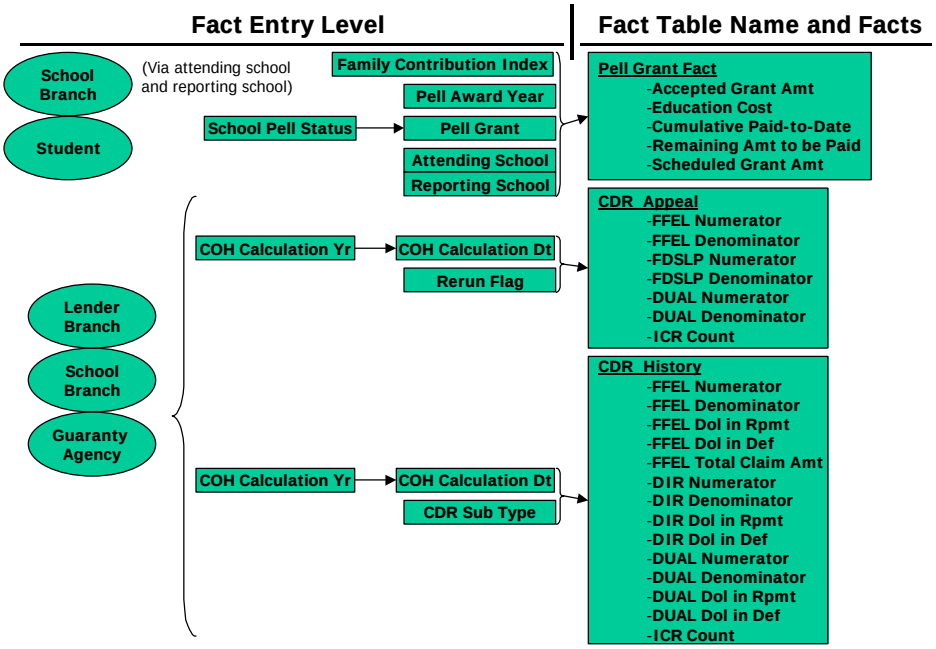
Facts

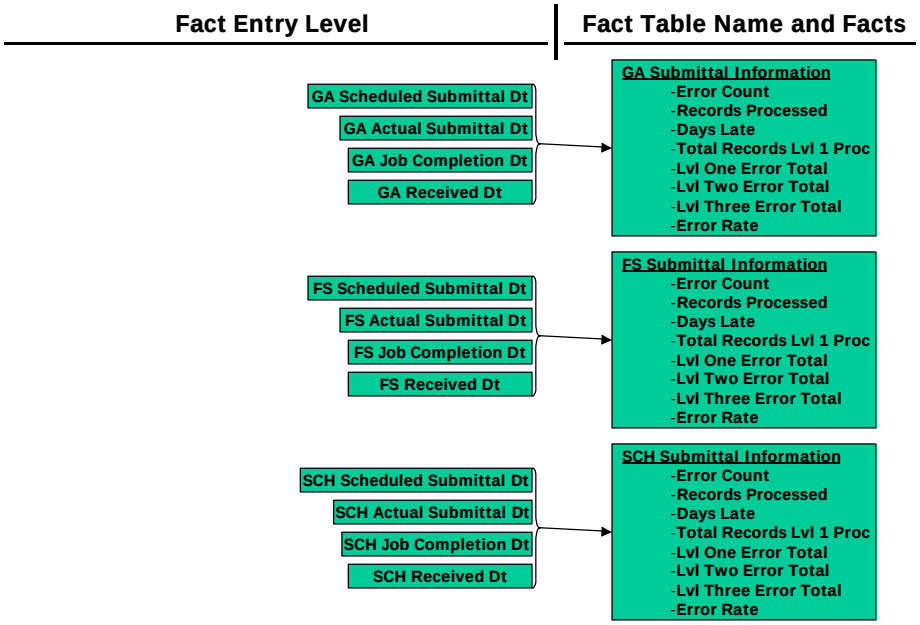
Each fact has an entry-level that defines the most basic level at which the fact is valid. For example, if a fact were on a table with only the year as the time representative, a month level calculation would not be permissible.




Fact Entry Level
Fact Table Name and Facts
Guaranty Agency
Student
Lender
Loan
GA Paid Date
Date Claim in Sys
Ins Claim Reason Cd
GA Paid Date
Cancellation Dt
Insurance Claim Payment
-Amount Diff
-Cumulative Amt
Insurance Claim Refund
-Cumulative Amt
Loan Cancellation
-Amount Diff
-Cumulative Amt
Lender Branch
Loan
Guaranty Agency
Student
School Branch
Lender Branch Servicer
Out Prin Bal Date
Out Int Bal Date
Loan Payment Dt
Loan Fact
-Loan Amt
-Out Int Balance
-Out Prin Balance
-Total Cancellation Amt
-Total Disbursement Amt
Loan Payments
-Payment Amt
-Principal Amt
-Interest Amt


Fact Entry Level
Fact Table Name and Facts
Lender Branch
Student
Loan
School Branch
Guaranty Agency
Lender Branch Servicer
Forgiveness Date
Forgiveness Type
Disbursement Date
Collection Date
Last Int Coll Date
Last Ins Cl Reason
Repayment Plan Date
Repayment Plan Type
Rehabilitation Ind
Repurchased Date
Refund Date
Adjusted Date
Loan Forgiveness
-Amount Diff
-Cumulative Amt
Loan Disbursement
-Amount Diff
-Cumulative Amt
Loan Supplement
-Total Reinsurance Claim Payment
-Current Supplemental Fee Balance
Loan Repayment Plan
-Term of Repayment in Months
Loan Repurchased
-Repurchased Amt
Loan Refund
-Amount Diff
-Cumulative Amt
Loan Unpaid Refund Discharged
-Amount Diff
-Cumulative Amt


Fact Entry Level
Fact Table Name and Facts
School Branch
Student
(Via attending school and reporting school)
Family Contribution Index
Pell Award Year
School Pell Status
Pell Grant
Attending School
Reporting School
Pell Grant Fact
-Accepted Grant Amt
-Education Cost
-Cumulative Paid-to-Date
-Remaining Amt to be Paid
-Scheduled Grant Amt
Lender Branch
School Branch
Guaranty Agency
COH Calculation Yr
COH Calculation Dt
Rerun Flag
CDR Appeal
-FFEL Numerator
-FFEL Denominator
-FDSLP Numerator
-FDSLP Denominator
-DUAL Numerator
-DUAL Denominator
-ICR Count
COH Calculation Yr
COH Calculation Dt
CDR Sub Type
CDR History
-FFEL Numerator
-FFEL Denominator
-FFEL DoI in Rpmt
-FFEL DoI in Def
-FFEL Total Claim Amt
-DIR Numerator
-DIR Denominator
-DIR DoI in Rpmt
-DIR DoI in Def
-DUAL Numerator
-DUAL Denominator
-DUAL DoI in Rpmt
-DUAL DoI in Def
-ICR Count


Fact Entry Level
Fact Table Name and Facts
GA Scheduled Submittal Dt
GA Actual Submittal Dt
GA Job Completion Dt
GA Received Dt
GA Submittal Information
-Error Count
-Records Processed
-Days Late
-Total Records Lvl 1 Proc
-Lvl One Error Total
-Lvl Two Error Total
-Lvl Three Error Total
-Error Rate
FS Scheduled Submittal Dt
FS Actual Submittal Dt
FS Job Completion Dt
FS Received Dt
FS Submittal Information
-Error Count
-Records Processed
-Days Late
-Total Records Lvl 1 Proc
-Lvl One Error Total
-Lvl Two Error Total
-Lvl Three Error Total
-Error Rate
SCH Scheduled Submittal Dt
SCH Actual Submittal Dt
SCH Job Completion Dt
SCH Received Dt
SCH Submittal Information
-Error Count
-Records Processed
-Days Late
-Total Records Lvl 1 Proc
-Lvl One Error Total
-Lvl Two Error Total
-Lvl Three Error Total
-Error Rate